特許庁アーキテクチャ標準仕様書

第1.1版

平成28年6月

特許庁

改訂履歴

項番	版数	作成日/改訂日	変更箇所	変更内容
1	1.0	平成27年7月31日	新規	
2	1.1	平成28年5月31日	はじめに （1） 表 （1)-1 本書に関係する標準・規約類の概要 項番9	『HTML文書規約』に関する当該行を削除。
3	1.1	平成28年5月31日	2.2 プロセス（ルールの適用工程）	SLCPプロセス番号を削除。
4	1.1	平成28年5月31日	3. ルール（規約・特例・設計指針・推奨）	ルールの目的を追加。各ルール（規約・特例・設計指針・推奨）の見直しに伴う修正。ルールの説明に曖昧さがある箇所を修正。ルールの理解を促進するための補足説明（図表含む）を追加。章構成の変更。
5	1.1	平成28年5月31日	3.1.1.3.1 システム構成要素の種類及び責務	配下の章節の記述順を修正。
6	1.1	平成28年5月31日	3.1.1.3.1.3 システム構成要素間のアクセスパス	3.3.3.2のアクセスパスの制限事項に関する記載を移動((2), (3))。アクセスパスの特例を追加((4))。システム構成要素の配置の特例及び特例となるシステム構成要素のアクセスパスについて追加((5))。3.3.3.4.2の個別データベースへのアクセスルールに関する記載を移動((6), (7))。アクセスパスをまとめた表を追加((8))。
7	1.1	平成28年5月31日	3.1.2 システムの動的な振る舞い	処理シーケンスの内容をサンプル集として分冊化（参考1）。
8	1.1	平成28年5月31日	3.1.2.1 APレイヤの定義及びコンポーネントとの関係 3.1.2.2.2 APレイヤ・コンポーネント定義 3.1.2.3.2 APレイヤ・コンポーネント定義 3.1.2.4.2 APレイヤ・コンポーネント定義	3.1.1配下の章節を移動。
9	1.1	平成28年5月31日	3.1.2.2.3 データの保護方式	3.1.2.6.2のBPMSのデータ授受に関する記載を移動。
10	1.1	平成28年5月31日	3.1.2.4.2 APレイヤ・コンポーネント定義	3.1.2.4.5のディレード処理管理情報に関する記載を移動。
11	1.1	平成28年5月31日	3.1.2.5 帳票出力方式	帳票出力機能の紙出力サブシステムへの集約化に伴い，本書で扱わない旨を追記。ルールから削除。
12	1.1	平成28年5月31日	3.1.2.6.1.4 外部システム連携層を介した外部システム連携	3.3.4のESB及び外部システム互換機能の特記事項を移動。
13	1.1	平成28年5月31日	3.2.1 インタフェースのルール	冒頭にインタフェースの種類に関する説明を追加。インタフェース名の命名ルールを分冊化（別冊1）。命名ルールの見直しに伴う修正。
14	1.1	平成28年5月31日	3.2.1.1.2 内部インタフェースのプロトコル	各システム構成要素が提供するインタフェースの見直しに伴う修正。
15	1.1	平成28年5月31日	3.3 多階層構造の層毎に遵守すべきルール	3.3.1～3.3.6の冒頭の層の責務，システム構成要素の責務の記載について修正。
16	1.1	平成28年5月31日	3.3.1 UI層及びプレゼンテーション層のルール	画面，帳票のユーザインタフェース設計に関する内容を参考情報として分冊化（参考2）。
17	1.1	平成28年5月31日	3.3.2 ワークフロー層のルール	BPMNの表記法に関するルールを分冊化（別冊2）。BPMN記載ルールの適用範囲の変更。BPMN要素の使用可否の変更。ビジネスプロセスパターン事例の追加。
18	1.1	平成28年5月31日	3.3.2.1.3 プロセスデータとして保持する情報	プロセスデータとして保持する情報の見直しに伴う修正。

項番	版数	作成日/改訂日	変更箇所	変更内容
19	1.1	平成28年5月31日	3.3.3 業務アプリケーション層のルール	特許庁開発フレームワークの内容をアプリケーション基盤機能の参考情報として分冊化（参考3）。アプリケーション基盤機能の利用に関するルールを追加。
20	1.1	平成28年5月31日	3.3.4 外部システム連携層のルール	ESBの利用を規約から推奨に変更。外部システム連携層が持つ機能の記載内容の見直しに伴う修正。
21	1.1	平成28年5月31日	3.3.5.1 共有データベースに配置するデータ及び及びアクセスルール	基盤APIの使用条件の修正。特定サブシステム間共有データに関するアクセスルールを追加。
22	1.1	平成28年5月31日	3.3.6 ビジネスルール管理層のルール	BRMSの呼び出し方式の選択ルールを「サービス方式」に一本化することに伴う修正。
23	1.1	平成28年5月31日	3.4 システム開発全般で遵守すべきルール	プログラムプロダクトに関するルールを分冊化（別冊3）。システム機能に関する共通的なルールを分冊化（別冊4）。特許庁システムが標準とする非機能要求グレードのルールを削除。
24	1.1	平成28年5月31日	3.4.1 データ設計に関するルール	用語管理のルール及び論理データベース設計の命名ルールを分冊化（別冊1）。用語の種類，命名ルールの見直しに伴う修正。異なるデータベース間でのテーブル名（及びカラム名）の重複と同音異義・異音同義語を禁止する範囲を修正。特別な構造を持つデータの取り扱い方法の章節を削除。

はじめに

(1) 本書の位置づけ

本書は,『特許庁アーキテクチャ策定指針[1]』を基礎として今後刷新される個別システムが準拠しなければならない標準的な構造を定義した文書であり, 特許庁アーキテクチャ策定指針の下位文書に該当する。

本書は個別システムがとるべき構造として, 基本的にアプリケーションのアーキテクチャを対象としたルールを策定するものであり, ハードウェアやネットワーク等といった設備やバックアップ等といった運用等については, 本書とは別に定める標準・規約類に従うこと。

以下に, 本書で定めるアーキテクチャに関係する標準・規約類について, 本書との関係と概要を示す。

表 (1)-1 本書に関係する標準・規約類の概要

項番	本書との関係	標準・規約類文書	概要
1	本書で定めるアーキテクチャ以外のルールであるが, 本書で定めるルールに関係が強く, 本書から参照しているもの。	『ポリシー実施手順(開発編 別紙「設計基準」)』	特許庁情報セキュリティーポリシーを遵守するための手順を定めたもの。
2		『LDAPアクセス運用』	特許庁システム内で提供されるLDAPサービス(共通テーブル管理システム)についての利用方法を定めたもの。
3		『運用管理エージェント登録ガイドライン』	特許庁システムの運用管理マネージャの管理対象として各システムを登録するための規約を定めたもの。
4		『バックアップ設計指針』	特許庁システムが扱うデータのバックアップの指針を定めたもの。
5		『特許庁システム リリースポリシー』	リリースに関わる作業と運用のルールを定めたもの。
6		『パッチ適用方針』	パッチ適用の判断基準や対策実施等に関する全体的な方針を定めたもの。
7		『日本国特許庁電子文書交換標準仕様 XML編』 『日本国特許庁電子文書交換標準仕様 特定書類編』	日本国特許庁内外で交換する電子文書のフォーマット等の仕様を定めたもの。
8	本書で定めるアーキテクチャに基づき, 特定システムに特化して詳細化したルールを定めたもの。	『DBアクセス基盤利用者向けガイドライン』	DBアクセス基盤サービスが提供する機能や, それを利用するシステムが従うべき指針を定めたもの。

特許庁アーキテクチャ策定指針及びその他の標準・規約類との関係については,『特許庁PMO標準・規約類における整備及び管理方針』を参照すること。

なお, 開発プロセスに関するものについては, 本書とは別に策定される『設計・開発ガイドライン(システム刷新&新規システム構築編)』を参照すること。

(2) 本書の利用者及び利用目的

本書は, 個別システム刷新に関係するステークホルダ(情報技術統括室職員, 特許庁PMO, システム利用者, 原課, 要件整理補助(支援)業者, 調達支援業者, 設計・開発ベンダ, システムインテグレーションベンダ, ハードウェアベンダ, オペレーションベンダ等)向けに作成されたものであり, 当該ステークホルダが本書を利用しシステムの構造を標準化するためのルールに従い個別システム刷新を行うことにより, 段階的刷新を通じ特許庁システム全体として統一された保守性と移植性の高いシステムを実現することを目的とする。

[1]『特許庁アーキテクチャ策定指針』 http://www.jpo.go.jp/torikumi/system/pdf/architecture/houshin.pdf

(3) 本書の文書構成

本書は，以下の章から構成される。

1章　設計方針

最適化計画で示されているシステムの多層構造化とデータの論理的な集約化による簡素化を基礎としつつ，更に特許庁アーキテクチャ策定指針に示された方向性で標準の具体化を進めるための設計方針を示す。

2章　ルールの考え方

設計方針に基づいて段階的に刷新される各刷新対象システムの構築に必要となる，設計に関与するステークホルダ（設計・開発ベンダ等）の遵守事項（ルール）を，以下の点において整理する。

- スコープ（ルールの適用範囲（システム））
- 強制力（ルールに含まれる設計に関与するステークホルダ（設計・開発ベンダ等）の裁量の範囲であり，規約，特例，設計指針，推奨の四つに分類）

3章　ルール（規約・特例・設計指針・推奨）

上述したルールの考え方に基づいて分類される設計に関与するステークホルダ（設計・開発ベンダ等）に課すルールを示す。具体的には，システムを構成するシステム構成要素やコンポーネント等の各要素の横断的な関係を，静的な構造，動的な振る舞いとしてモデル化し定めたルール（3.1章）と，各要素における個別ルール（3.2章～3.4章）を示す。

①　技術方式において遵守すべきルール（3.1章）

技術方式を静的な構造と動的な振る舞いの観点からモデル化し，当該モデルに基づきルール化する事項を規定する。具体的には，静的な構造として多階層構造をモデル化し，システム構成要素の責務，各階層への配置，依存関係，システム構成要素の論理構成（ソフトウェア構成）等をルールとして規定する。また，各システム構成要素の内部を構成するコンポーネントも同様に，APレイヤ・コンポーネントにおける静的な構造や動的な振る舞いをモデル化し，ルール化する事項を規定する。

②　多階層構造の層の境界において遵守すべきルール（3.2章）

サブシステム間や多階層構造の層の境界において遵守すべきルールを規定する。

- インタフェースのルール

③　多階層構造の層毎に遵守すべきルール（3.3章）

サブシステムを構成する多階層構造の層毎に遵守すべきルールを規定する。

- UI層及びプレゼンテーション層
- ワークフロー層
- 業務アプリケーション層
- 外部システム連携層
- 基盤機能層及びデータベース層
- ビジネスルール管理層

④　システム開発全般で遵守すべきルール（3.4章）

3.2章，3.3章に該当しないシステム開発全般で遵守すべきルールを規定する。

- データ設計
- 文字コードの扱い
- 開発言語の選定

2章に記載した考え方に基づく3章との関係を概念的に図示化したものを以下に示す。

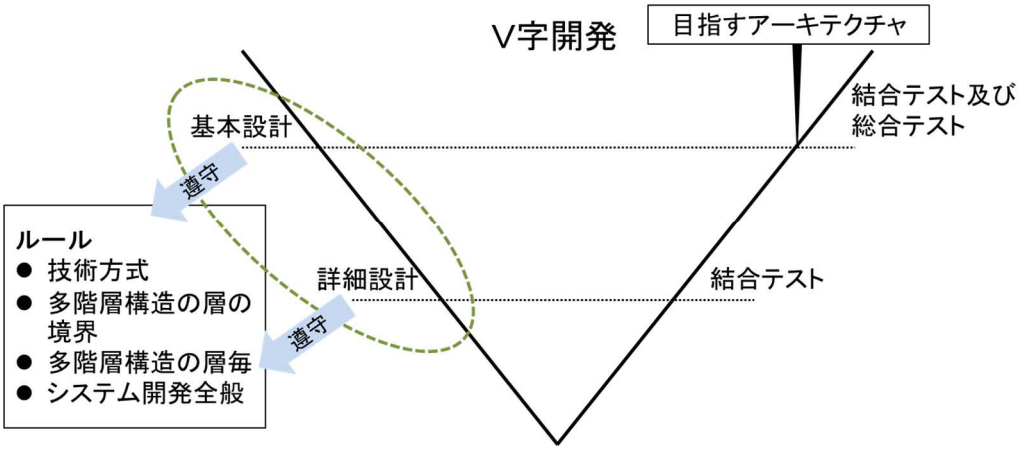


図 (3)-1 2章と3章の関係概念図

別冊1 命名ルール編

本書の各要素における個別ルール(3.2章～3.4章)のうち，用語のルール及び名称の標準化のための命名ルールに関する内容を規定する。

別冊2 BPMN記載ルール編

本書の多階層構造の層毎に遵守すべきルール(3.3章)のうち，BPMNの表記法に関する内容を規定する。

別冊3 プログラムプロダクト編

本書のシステム開発全般で遵守すべきルール(3.4章)のうち，プログラムプロダクトに関する内容を規定する。

別冊4 システム機能共通編

本書のシステム開発全般で遵守すべきルール(3.4章)のうち，特許庁システムの機能に関する共通的な内容を規定する。

別紙1～3 特許庁アーキテクチャ標準仕様書適合確認チェックリストの雛形

刷新される各個別システムの設計内容が，上述のルールに適合しているかを確認するために使用するチェックリストの雛形である。別紙1，別紙2，別紙3にそれぞれ特許庁アーキテクチャ標準仕様書，別冊1，別冊2のルールのチェックリストを掲載する。

参考1 処理シーケンスサンプル集

本書の技術方式において遵守すべきルール(3.1章)のうち，APレイヤ・コンポーネントにおける動的な振る舞いを処理シーケンスとして定める際に参考とするサンプル集を掲載する。

参考2 画面・帳票編

本書の多階層構造の層毎に遵守すべきルール(3.3章)のうち，画面や帳票のユーザインタフェース部分を設計・開発する際に定めておくべき方針の具体例を，参考情報として掲載する。

参考3 アプリケーション基盤編

本書の多階層構造の層毎に遵守すべきルール(3.3章)のうち，アプリケーション基盤を設計・開発する際に定めておくべき方針のの具体例を，参考情報として掲載する。

(4) 本書の利用方法

本書の利用者及び利用方法について以下に示す。

表 (4)-2 本書の利用者及び利用方法

(○:利用する, ー:利用しない)

利用者 / 利用方法	情報技術統括室	特許庁PMO	システム利用者, 原課	要件整理補助業者, 調達支援業者	設計・開発ベンダ	システムインテグレーションベンダ	ハードウェアベンダ	オペレーションベンダ
将来像のイメージ共有	○	○	○※	○	○	○	○	○
システム構造の定型化(ルールの理解・遵守)	○	○	○※	○	○	○	ー	ー
技術的整合性確保(コントロール及びチェック)	○	○	ー	○	○	ー	ー	ー

※ルールに従い画面設計等の設計レビューに関与するために必要。

上記利用方法のうち「将来像のイメージ共有」として本書を利用する場合は, 1～3.1章を主に参照することが望ましい。特に,「3.1 技術方式において遵守すべきルール」については, 目指すアーキテクチャが図示されている。

「3.2 多階層構造の層の境界において遵守すべきルール」,「3.3 多階層構造の層毎に遵守すべきルール」,「3.4 システム開発全般で遵守すべきルール」は, 詳細なルールを記載しているため, 必要な箇所をその都度参照してルールを確認するといった利用方法を想定している。

(5) 本書の運用方法

本書の運用方法について以下に示す。

① 運用開始時期

平成28年6月から運用を開始する。

② 改定時期

平成29年3月末, 平成30年3月末及び平成31年3月末の3回の時期において改定を予定している。

③ 整備及び管理

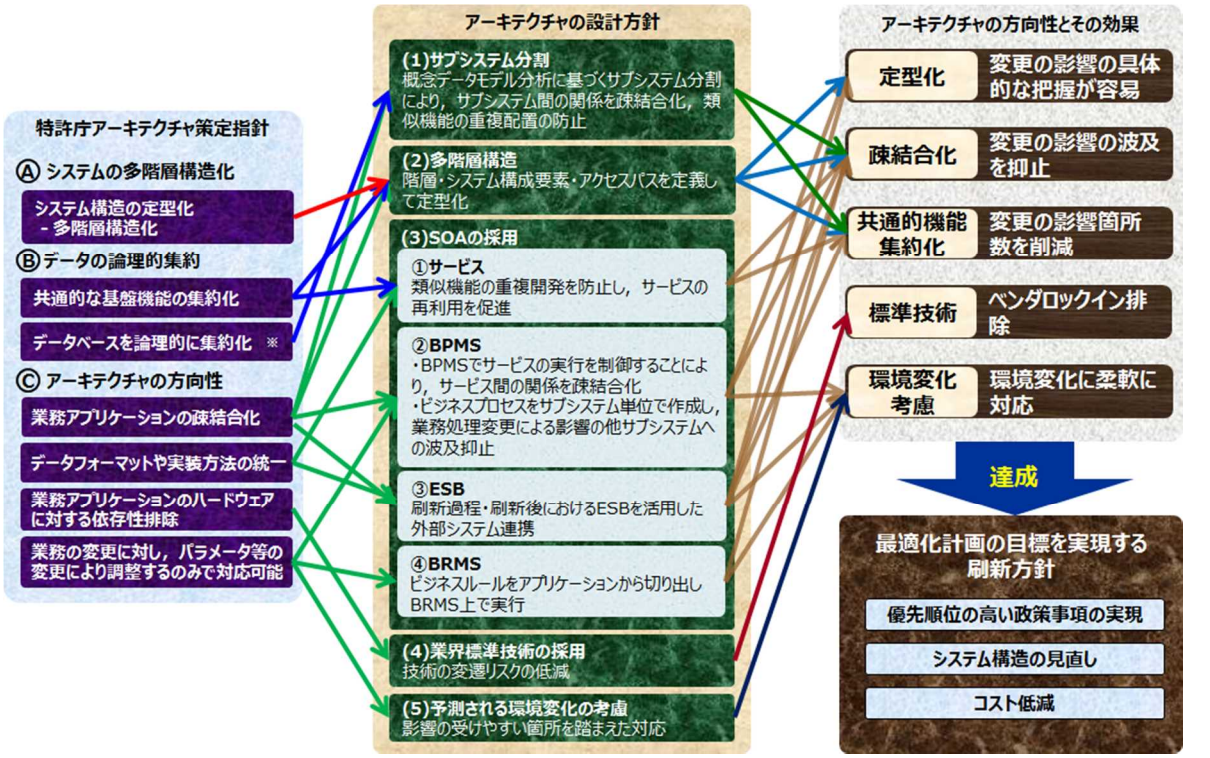
『特許庁PMO標準・規約類における整備及び管理方針』に従う。

－ 目 次 －

1. 設計方針

1.1 アーキテクチャの設計方針の導出

特許庁アーキテクチャ策定指針に示されたシステムの多層構造化・データの論理的な集約化による簡素化・アーキテクチャの方向性とアーキテクチャの設計方針の対応関係，アーキテクチャの設計方針から得られる効果及び最適化計画の目標を実現する刷新方針の達成について，下図に整理する。



※ データの論理的集約化は「データ統合方針書」にて具現化する。

図 1.1-1 アーキテクチャの設計方針の導出

特許庁アーキテクチャ策定指針で示されるシステム構造の見直しの考え方，アーキテクチャの方向性を受け，システム構造の簡素化を以下に示す(1)～(3)により実現する。

(1) サブシステム分割
(2) 多階層構造
(3) SOAの採用

また，以下に示す(4)及び(5)により，(1)～(3)で実現した簡素化したシステム構造を長期的に利用可能にする。

(4) 業界標準技術の採用
(5) 予測される環境変化の考慮

このようなアーキテクチャの設計方針を採用することにより，システム構造の複雑化やデータの個別システムへの分散といった現状の問題点を解決する。

次項にてアーキテクチャの設計方針の詳細について述べる。

1.2 アーキテクチャの設計方針の詳細

以下にアーキテクチャの設計方針の詳細を述べる。

(1) サブシステム分割

特許庁システム全体を，設計・開発や管理の面における適切な単位となるよう論理的に分割し，サブシステムとして定義する。各サブシステムは，可能な限り互いに疎であることが望ましい。疎であるとは，互いの関連性が低く，連携する頻度や受け渡す情報の量が少ない状態を指す。疎の関係となるよう分割することにより，変更によるシステムの改修範囲を個々のサブシステム内にできる限り局所化することを可能とする。本設計方針においては，既存システムの枠にとらわれずに概念データモデルの観点と法律の観点から，業務を同じ目的の業務処理の集合に分割する。

また，最適化計画の方針に従い，各業務システムに分散していた重複データを論理的に共有データベースに集約して管理[2]するにあたり，それ以外の，各業務システムにおいて個別に保有し管理すべきデータを個別データと定め（詳細については『データ統合方針書』を参照のこと。），所定の業務システム内に隠蔽し他の業務システムからのアクセスを禁止することでデータの更新整合性[3]を確保しやすくする。

当該個別データを隠蔽する業務システムとして，上記業務処理単位に分割したサブシステムを採用する。

更に，将来起こりえる環境変化に対してもシステムの改修範囲が大きくなることを防ぐため，特許庁システムにおいて最も大きな環境変化の要因である法律の観点を考慮した分割とする。特許法，実用新案法等，特許庁業務の根拠法令の種類と，特許庁業務のシステムを利用した実現方法を示した特例法の双方を踏まえることで，特許庁業務を疎結合かつ適切な粒度で分割したサブシステムを採用する。

今後，新規に業務が追加された場合においても，概念データモデルのメンテナンス及び法律の観点から業務の位置づけを整理し同様のサブシステム分割を行い，どのサブシステムに業務を追加するか，あるいは新規にサブシステムを追加するか等といった検討を行うものとする。

(2) 多階層構造

分割した各サブシステムの構造を定型化するために，最適化計画の方針に従い，多階層構造を定義し，多階層の各層に配置するシステム構成要素とシステム構成要素のアクセスパスを定める。定義した多階層構造に対し，以下に示す設計方針を適用することにより，更なる定型化と疎結合化を行い，サブシステムの保守性・移植性を向上させる。

- 各層の責任分担を適切に定め層毎に機能を集約することにより，改造が必要になった場合における影響箇所の特定の容易化。
- 層と層のアクセス方法を標準化するとともに層間の依存関係を統制することにより，改造が必要になった場合における影響が他の層に拡散することを抑制。

(3) SOAの採用

SOAは2000年初頭から広まった技術であり，SOAを活用してレガシーシステムを流用しつつシステム刷新を行った実績も数多い。特許庁における最適化計画の刷新方針において，SOAの採用は以下に示す利点を有する。

- 最適化計画において掲げられたシステム構造の見直しの「業務APを疎の関係とすること」との親和性が高い点。
- 既存システムを維持しつつ段階的に個別システムを刷新することとの親和性が高い点。

---

[2] このような特徴を持つサブシステムは，「表 3.1-4 サブシステムの分類とシステム構成要素」の共有データベース管理サブシステムに分類される。共有データベース管理サブシステムは特許法や条約に基づく法域をベースした分類の単位で分割されるため，複数存在し得る。

[3] 例えば，決裁前の処分情報を後続工程の業務システムが参照してしまうことにより，誤って決裁されたものとして後続の業務システムが処理を実行するパターン（チェックアウト中の情報参照によるデータ不整合）や，外注中の案件に対して誤って処理を実行するパターン（ロングタームトランザクション中のデータ不整合）等。共有データベースに業務として確定した最新情報を集中化するとともに，共有データベースの情報又は自システム内のCRUDを制御できる個別データベースの情報のみを用いて各業務システムが処理の実行及び個別データベースの更新を実施することにより，データの整合性を担保可能とする。

- 最適化計画が10年にわたる極めて長期的な計画であることとの親和性が高い点（技術動向が比較的安定している業界標準のアーキテクチャである点）。

したがって，サブシステム分割や多階層構造を適用したシステム構造を支える技術としてSOAを採用する。

SOAの考え方に基づき，特許庁システムの機能をサービスで構成し，各サブシステムから利用する。特許庁システムのサービス間の連携を担う技術として，BPMSやESBを適用する。また，変更の頻度の高いビジネスルールをアプリケーションから切り出して実行するためにBRMSを使用する。具体的な方針を以下に示す。

A. サービス

SOAの考え方を適用し，特許庁システムの機能をサービスとして構成する。サービスは共通的に利用することを想定し，サービスインタフェースの仕様を管理する。これにより，サービスの提供する機能の再利用が促進され，類似機能が重複して開発されることを抑止する。

また，サービスへアクセスする方法として業界標準の通信プロトコルを使用する。これによりサービスへの依存関係を疎結合にし，インタフェースが変わらない限り，サービスの内部構造が刷新等により変更された場合においても，その影響が利用者側に波及しないようにする。

B. BPMS

BPMSは，サブシステムを構成するサービスの実行順序や実行条件の制御を一元管理する役割を持つ。これにより，サービス同士の直接的な連携を排除し，サービス間の関係を疎結合に保つことを可能とする。

BPMS上で実行されるビジネスプロセスのアクティビティは業務として意味のある最小の単位で記述する。また，ビジネスプロセスはBPMNで記述し，そのアクティビティはサービス実行の単位と対応付けられるものである。これにより，BPMNで業務処理の流れが可視化されることに加え，業務的に理解できる要素とシステム上対応するアプリケーションの処理が対応付けられるため，業務の変更に伴うシステムへの影響範囲の把握を容易にする。また，業務の要件整理（記述モデル），業務可視化資料（分析モデル），システム可視化資料（実行可能モデル）の三つを同じ記法（BPMN）で作成することにより，ワークフローの管理負担の軽減や，可読性の向上が期待できる。

また，ビジネスプロセスはサブシステムの単位で作成する。サブシステムの範囲は業務的な責務の境界であるため，業務処理を記述するビジネスプロセスやビジネスプロセスがアクセスするサービスは，同じサブシステム内に閉じて実行されるものを多く含むことが想定される。このようにビジネスプロセスの単位を定めることで，業務処理の変更の際に，他のサブシステムに対して大きな影響が波及することを防ぐ。

なお，業務の実行制御方式として，単純にアプリケーションを順次実行する処理方式と比較すると， BPMSはビジネスプロセスの進行状況を管理する仕組み上，性能面においてビジネスプロセスやアクティビティの粒度の設計に注意を要する。しかし，特許庁アーキテクチャ策定指針において高めるべき品質特性として特に保守性が掲げられているところ，上述のとおりBPMSの適用によって保守性の向上が期待され，また，他のビジネスプロセス管理技術と比較した場合，ビジネスプロセスの可視性が高く，HTTP(S)のようなWebサービスの標準規格にも対応している等の優位性があることからBPMSを採用することとする。

C. ESB

外部システムへのアクセスについては，ESBが標準化された内部システムのプロトコルでアクセス可能なサービスインタフェースを提供する。またESBには，外部システムが利用する様々な通信方式とのギャップを吸収する役割を持たせる。これにより，外部システムとの連携であっても標準化された内部インタフェースのようにアクセスでき，内部システムのシステム構造の定型化を促進するほか，外部インタフェースアクセスの変換機能を集中化し，類似の機能が各サブシステムで重複して開発されることを防ぐ効果や，外部システムと疎結合を保ち，外部システムの刷新や変更による大きな影響が内部システム側に波及されることを防止する効果を期待するものである。具体的には，外部システムが刷新された際に，内部システムのアプリケーションを極力再度修正しなくてもよいようにすることを可能とする。

外部システムが提供するインタフェースと内部システムで使用されるインタフェースとの変換においては，ESB製品によって実現が困難となるプロトコル変換や処理単位（単件，複数件）の変換等が想定されているため，補完的に開発する外部システム互換機能を配置し，それらが協調動作してこの役割を担うものとする。外部システム互換機能は，アプリケーション，プログラムプロダクト又はそれらの組み合わせにより実現する。

外部システムへのアクセス方式として，ESB等のプログラムプロダクトを使用せず，アプリケーションのみで実

現するという方式も考えられるが，アプリケーションの作り込みを極力避け，ESB製品の設定情報の変更等の対応により保守性・移植性・開発効率性が高まる効果や，特許庁システムのToBeアーキテクチャがESBで一元化されることによる定型化の促進や効率的なリソースの活用，期待される統制面の向上といった効果を重視する。また，他の外部システムとの連携技術と比較した場合，異種のシステム間の接続に広く一般的に使用されており，サービス間の複雑な連携や性能面の拡張性の高さ，Webサービスの標準規格にも対応している等の優位性があることからESBを採用することとする。

D.　BRMS

業務処理の中から変更の頻度の高い業務をビジネスルールとしてアプリケーションから切り出して実行するために使用する。これにより，ビジネスルールの変更や追加の影響が局所化される。BRMSはビジネスルールをデシジョンテーブル等の形式で宣言的な表現で記述できるため，ビジネスユーザでも比較的理解しやすく，変更による影響箇所が特定しやすいビジネスルール管理技術である。

(4)　業界標準技術の採用

特許庁システムの業務を踏まえつつも，業界で広く一般的に使用されている技術に準拠し，プログラムプロダクト毎に機能要件や仕様を標準化する。このことにより，技術の変遷リスクを低減するとともに特定ベンダの製品の機能に依存しないようにする。

なお，複数のプログラムプロダクトを統合した，いわゆるスイート製品でなければ実現できないようなアーキテクチャを本書で定めることはないが，本書に示すルールに従う限り，そのような製品群を排除するものではない。例えば，多階層の各層に配置するシステム構成要素群を一つのスイート製品で実現してもよい。

(5)　予測される環境変化の考慮

過去の制度改正を踏まえて，制度改正等によって影響の受けやすい箇所を特定し，アーキテクチャ上の対策をとることにより，環境の変化によるシステムの影響範囲を極小化する。

2. ルールの考え方

本書の設計方針に従い刷新対象システムを構築するにあたり，刷新対象システムの業務特性に応じた要求仕様を考慮した設計が必要である。しかしながら，産業財産権を付与し当該権利を維持するという特許庁の事務処理業務には，出願から権利消滅に至るまでに業務的な手続き（受付，審査，登録等）の流れが存在するという点や，業務的な手続きにより出願に係る業務データの状態が変化し，業務データの状態によって実施されうる手続きが管理されるという点で共通の特徴があるため，本書の設計方針に基づきシステム構造を定型化することが効果的な部分も存在している。すなわち，システム構造の定型化のために遵守させる強制力の強いルールもあれば，設計の裁量を残す強制力の弱いルールも存在する。また，ルールを適用するシステムもあれば，ルールを適用しないシステムもある。

また，設計に関与するステークホルダ（設計・開発ベンダ等）の設計ノウハウを最大限に活かすことが肝心であるものの，システム構造を定型化するための何かしらの制限も必要である。

以上の点を考慮し，以下，本書の設計方針に基づいてシステム構築を行うために必要となるルールの考え方について，スコープ（ルールの適用範囲）及び強制力（ルールの裁量）の観点について整理する。

2.1 スコープ（ルールの適用範囲）

本書ではアプリケーションのアーキテクチャを対象としたルールを策定する。ハードウェアやネットワーク等の設備等については基本的に対象外であるが，アプリケーションのアーキテクチャを定めるために関係するものについてはその限りではない。

また，本書で定めるルールは，産業財産権の付与及び維持に関する特許庁の事務処理システムに適用する。適用されるシステムの具体例について，平成25年度時点の既存システム等に対してマッピングしたものをToBe対象システムとして『全体システム概念設計書』の「1章」にまとめているので参照のこと。

本書で定めるルールは，全てのToBe対象システムに適用するものであるが，システムの特性や条件によって適用範囲が異なる。以下に詳細を述べる。

(1) 階層定型化サブシステム

特許庁の事務処理システムのうち，産業財産権法に基づく事務処理を行うシステムについては，上記のとおり事務処理内容に共通事項が多い。このようなシステムについては，階層定型化サブシステムとして，システム間のインタフェースやシステム内の構造に対し，本書にて定めたルールを適用してシステム構造を定型化する。その他の定型化による効果が見込める事務処理システムについても階層定型化サブシステムに分類する。

なお，階層定型化サブシステムに関する詳細な説明については，「3.1.1.3.1.1 サブシステムの分類に基づくシステム構成要素」を参照すること。

(2) インタフェース定型化サブシステム

特許庁の他の事務処理業務と共通事項のない特殊な責務を担う事務処理システムについては，個別の要求仕様を考慮し，個別にアーキテクチャを定めるシステムとして，システム構造の定型化を部分的に留める。

具体的には，階層定型化サブシステムとのインタフェースやプログラムプロダクトの適用に関するルールを定めるに留め，責務に特化した製品や技術方式を適用するアーキテクチャの採用を許容する。

本システムの具体例を「表 2.1-1 インタフェース定型化サブシステムの例」に示す。「表 2.1-1 インタフェース定型化サブシステムの例」に示したもの以外のシステムについては，設計に関与するステークホルダ（設計・開発ベンダ等）がシステム固有の要件に基づき，特許庁と協議の上，インタフェース定型化サブシステムへの分類可否を決定する。

なお，インタフェース定型化サブシステムに関する詳細な説明については，「3.1.1.3.1.1 サブシステムの分類に基づくシステム構成要素」を参照すること。

表 2.1-1 インタフェース定型化サブシステムの例

項番	サブシステム	システムID	理由	刷新前システム
1	外部システム連携	－ （新規）	個々の外部システムとの連携でそれぞれ使用されるプロトコルに対応した製品や，その他のギャップを吸収するための仕組みを有するため。	－
2	情報活用系	SY47(*1)	特許庁におけるデータウェアハウスの機能を一般的な製品で実現可能であることが見込まれるため。	*1：データウェアハウスシステム
3	リソース管理	SY45(*2)	LDAPサーバ機能を一般的な製品で実現可能であることが見込まれるため。	*2：共通テーブル管理システム
4	運用監視	SY58	ジョブ管理やシステム監視といった機能は一般的な製品で実現可能であることが見込まれるため。	－
5	対外連携	SY33 SY33-2 SY35-2 SY43 SY05 SY44 (*3)	連携先システムに応じた連携方法を実現するものであり，定型化することはできないため。	*3：優先権証明書交換システム，外国包袋参照システム，照会システム（海外／一般ドシエ），電子現金納付システム，DE料管理システム，早期管理情報システム
6	ビジネスルール管理	－ （新規）	ビジネスルールの実行を一元的に行うといった特殊な責務を担うシステムであるため。	－
7	紙出力	－ （新規）	オンライン要求（フロアプリンタ出力），バッチ要求（センタプリンタ出力）等，帳票の出力制御を統合的に管理する責務を担うシステムであるため。	－

上記のうち，外部システム連携サブシステムとビジネスルール管理サブシステムは，ToBeアーキテクチャ上新規に追加されたシステム構成要素を含むという，特徴的な性質を持つ。

外部システム連携サブシステムについては，外部要因による変更の影響を局所化するために，外部システムへのアクセス方法とアクセスの際に生じる内部インタフェースとのギャップの吸収の方針をルール化の対象とする。

ビジネスルール管理サブシステムについては，ビジネスルールの可読性向上や変更による影響箇所の特定を容易にするために，ビジネスルールの記述ルールや適用対象をルール化の対象とする。

なお，外部システム連携サブシステムとビジネスルール管理サブシステムは，段階刷新においては複数のサブシステムで構成される可能性がある。

2.2 強制力（ルールの裁量）

ルールの強制力（裁量）を定めるにあたり，全てのルールに共通する目的は，以下のとおりである。

- 設計に関与するステークホルダ（設計・開発ベンダ等）が設計方針に従って設計を行うことにより目指すべきアーキテクチャを実現すること。

上記目的を達成するために，ルールに対して以下の(1)～(4)に示す四種類の強制力（規約，特例，設計指針，推奨）を定める。また，強制力の関係やルールの運用に係る補足を以下の(5)～(7)に示す。

(1) 規約

規約とは，ルールを適用する設計対象で選択し得る設計内容の中から定められた設計内容を採用するため裁量の余地はないものを意味する。業界標準技術であり，かつライフサイクルも極めて長い等の理由により，設計の裁量の余地を残す必要性がない設計対象については，本書において設計方針に従い導出したベストプラクティスに従うことをルール化する。

規約には，以下に示す二種類が存在する。例とともに示す。

- サブシステム全体で共通的に設計する対象について規約を示すもの
  例： システム構成要素間のアクセスは，表に示すアクセスパスに従うこと。

- 個々の設計によって決定する設計要素について規約を示すもの
  例： 業務アプリケーション（システム）のプロトコルはHTTP(S)とすること。

(2) 特例

特例とは，ルールからの逸脱（(6)で後述）をした設計内容のうち今後の開発においてもそのような設計が行われることが想定される場合に，妥当性が認められた条件とともにルールとして明文化したものであり，いわば判例のような情報として示すものである。特例である設計内容を採用する場合，設計・開発ベンダはその設計内容が特例の条件に合致することを情報技術統括室に説明し，特例として認められることが妥当であるかを協議の上判断するものとする。以下に例を示す。

例： 下位層から上位層のシステム構成要素へのアクセスは原則しないものとする。ただし，別途定める特異なケースについてはそのようなアクセスを特例として許容する。

特例は，実際のシステム構築で逸脱を許容した設計内容のフィードバックを精査し，以下に示す性質を持つものとする。

- 技術的な問題が解決したり回避策が登場することにより，将来的に特例とする必要性が無くなることが期待される設計内容
- 事例が充実し，逸脱となる条件が十分に一般化され整理でき，将来的に規約や設計指針となることが期待される設計内容

(3) 設計指針

設計指針とは，ルールを適用する設計対象で選択し得る設計内容の中から何を採用するかを示す指針であって，本指針に従う限りにおいて，設計内容に裁量の余地が残されているものを意味する。裁量の余地を残す理由は以下のとおりである。

- 業務特性に応じた要求仕様を満たす設計を可能とするため。
- ライフサイクルの比較的短いものや未定のもの（例えば，プログラムプロダクトや開発言語）に対して設計の時点におけるベストプラクティス（最も適切な設計内容）を採用可能とするため。
- 本書において引用している標準・規約類において設計内容の裁量の余地を残しているため。

設計指針には，以下に示す二種類が存在する。例とともに示す。

- サブシステム全体で共通的に設計する対象についての指針を示すもの
  例： BPMS製品は，次の機能を備える製品を採用すること。

- 個別の設計によって決定する設計要素についての指針を示すもの
  例：ワークフローのアクティビティの粒度は，業務の意味のある最小単位とすること。

なお，裁量の余地が残されているとはいえ，設計内容がベストプラクティスであることの立証責任は設計・開発ベンダにあるため，設計指針に従いどのような根拠により当該設計内容としたのかを情報技術統括室に説明すること。

(4) 推奨

推奨とは，設計指針及び当該設計指針に従い導出したベストプラクティスを例示するものの，設計指針に従い別の設計内容を採用するといった，裁量の余地を残すものを意味する。以下に例を示す。

例：フォントの種類及び文字の大きさは極力指定せず，OSやブラウザ等で設定された標準値を利用すること。

仮に，推奨されるベストプラクティスを適用しない場合は，設計・開発ベンダはその根拠を情報技術統括室に説明すること。

(5) 四種類の強制力の関係

上述の四種類の強制力のうち，規約及び設計指針は，独立して存在し得る強制力である。これら強制力は，後者が一定の裁量を認める強制力であるのに対し，前者は裁量を認めない強制力であることから，排他的な関係となるため，ルールを適用する設計対象毎にいずれか一方の強制力が適用される。

上述の四種類の強制力のうち，推奨及び特例は，単独で存在できない強制力であり，規約又は設計指針に従属する強制力となる。

四種類のルールの関係を下図に示す。

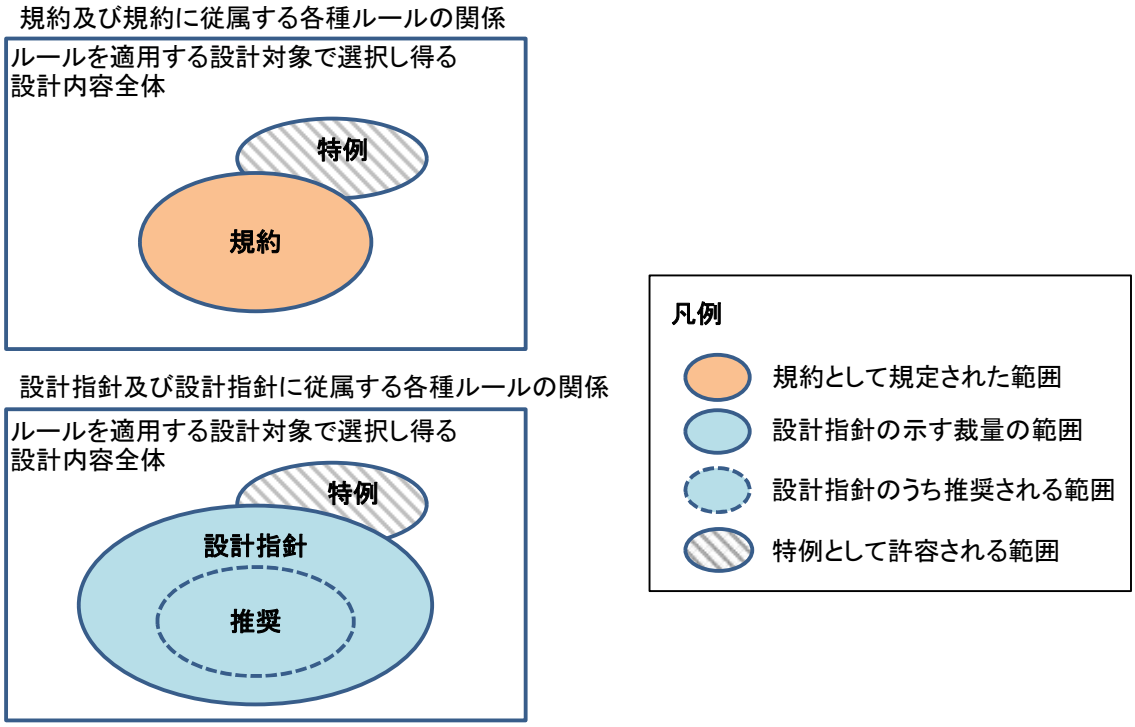
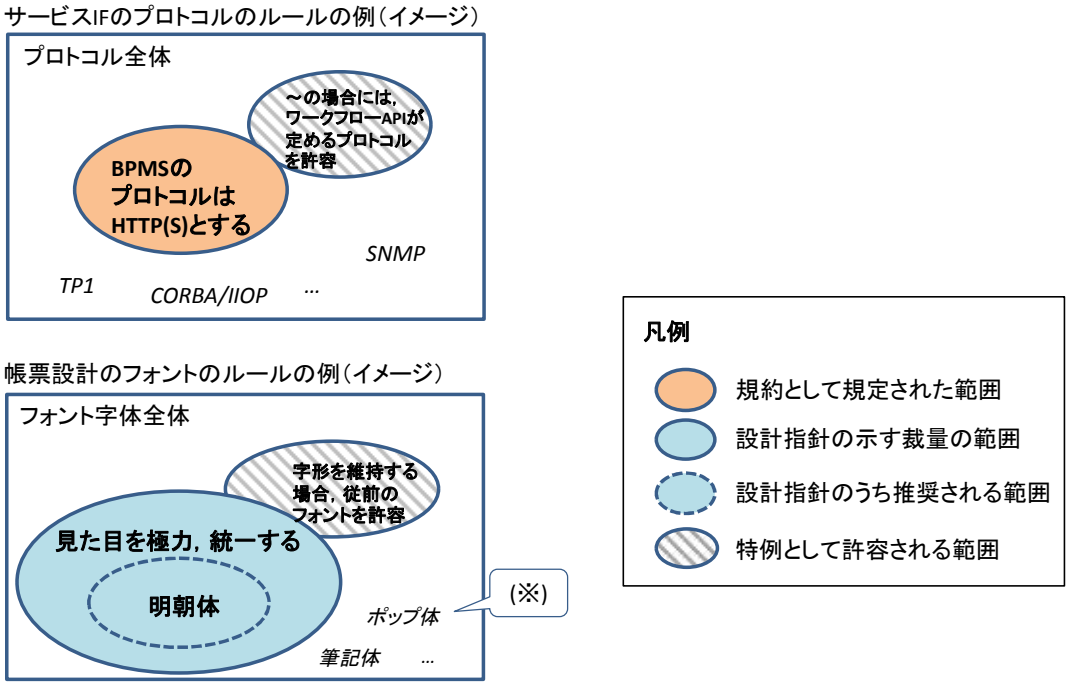


図 2.2-1 ルールの種類の関係



（※）ポップ体，筆記体が，特例（字形を維持する場合に従前のフォントとして使用する）に該当しない場合の例。
プロトコルの例も同様にTP1，CORBA/IIOP，SNMPが特例に該当しない場合の例。

図 2.2-2 ルールの関係の具体例

(6) ルールの逸脱の扱い

これまで述べた規約や設計指針等は特許庁アーキテクチャを実現するために定めたルールであり，原則として遵守することを必須とする。しかしながら，要件を満たすシステムを実現する上で，規約や設計指針をどうしても遵守できない，又は遵守することにより過度に複雑化する等の大きなデメリットが発生する場合があるため，そのような条件に該当すると認められる場合は，ルールで定めた設計内容の範囲からの逸脱を限定的に許容する。

規約及び設計指針からの逸脱は，あらゆる設計内容を許容するものではない。設計・開発ベンダがその逸脱する根拠と設計内容の妥当性を情報技術統括室に説明し，協議の上判断するものとする。

(7) 留意点

各ルールに対する設計内容の裁量の範囲の妥当性については，実際のシステム構築からのフィードバックの蓄積により確実なものとなる。したがって，システム構築における様々なリスクを考慮し，確実に規約化できるものを除き設計指針又は推奨として規定する。そして，設計からのフィードバックに基づき各ルールの裁量の範囲を見直し，必要に応じて本書を改定することとする。

3. ルール（規約・特例・設計指針・推奨）

本章におけるルールの表記方法を以下に示す。

- ルールの前段に「目的」（ルールの目的）を記載することにより，ルールを遵守することで達成したい事柄を明確化する。
- 規約及び設計指針は連番を付与し列挙する。また，設計指針はルールの表記上，「指針」と記載する。
- 推奨や特例ルールも連番を付与し，付随する規約又は設計指針のルールの直後に字下げして記載する。

ルールの表記例を以下に示す。

なお，表記例における「スコープ」の詳細については「2.1 スコープ（ルールの適用範囲）」を，「規約・特例・設計指針・推奨」の詳細については「2.2 強制力（ルールの裁量）」を参照のこと。

目的:　　XXXXXXXXX スコープ: XXXXXXXXX 規約1:　XXXXXXXXX 規約2:　XXXXXXXXX 　特例1: XXXXXXXXX 指針1:　XXXXXXXXX 　特例1: XXXXXXXXX 　特例2: XXXXXXXXX 　推奨1: XXXXXXXXX

3.1 技術方式において遵守すべきルール

3.1.1 システムの静的な構造

本章に示す技術方式のルール(規約, 特例, 設計指針, 推奨)は, 以下の章に示す。

- 3.1.1.3 多階層構造の詳細

3.1.1.1 多階層構造を採用する目的

特許庁システムにおいては, システムの静的な構造としてシステム構成要素を各層に配置し, それぞれのシステム構成要素間のアクセスパスを規定した多階層構造を採用する。

多階層構造を採用することで, 設計・開発における成果物(アプリケーション, プログラムプロダクト)が, 一つの巨大なプログラムではなく, 分割されて責務に応じたいずれかのシステム構成要素に関連付けられ, 分割されたプログラム間の依存関係が制限される。

これにより, 以下に列挙する要求を達成することを目的とする。

- 各層に配置したシステム構成要素の責務を適切に定めることにより, システム構成要素に改造が必要になった場合における影響箇所の特定容易化。
- 依存関係を制限したことによる, システム構成要素に改造が必要になった場合における影響箇所の拡散の極小化。
- 各層において利用するプログラムプロダクトの仕様を標準化することにより, プログラムプロダクトの交換容易性の向上。
- 個別システムの構造を画一化することにより, その後の『データ統合方針書』に基づくデータの論理的な集約化の促進。

3.1.1.2 多階層構造の概要

3.1.1.2.1 層の種類及び責務

層の種類及び各層に割り当てられた機能(責務)を下表に示す。なお, 各層の責務の詳細は「3.3 多階層構造の層毎に遵守すべきルール」を参照のこと。

表 3.1-1 層の種類及び責務

項番	層	責務
1	UI層	ユーザに対して入出力を伴う直接的なインタフェース(GUI)の提供
2	プレゼンテーション層	UI層とワークフロー層・業務アプリケーション層との連携
3	ワークフロー層	業務アプリケーションの連携をビジネスプロセスとして管理及び可視化
4	業務アプリケーション層	業務処理の実行
5	基盤機能層	データベース層へのアクセス
6	データベース層	特許庁システム全体で共有するべきデータの管理
7	外部システム連携層	外部システムが利用する様々な連携方式・通信方式とのギャップの吸収
8	ビジネスルール管理層	ビジネスルールの管理, 実行

3.1.1.2.2 層間の関係

層間には以下に列挙する関係を持たせる。

- UI層からデータベース層まで(「表 3.1-1 層の種類及び責務」の項番1から6)の間の層については階層化する。
- 外部システム連携層については，外部システムを様々な内部システムの層としてエミュレートする特許庁システム全体でただ一つの責務を持つため階層化の対象外とする。
- ビジネスルール管理層については，BRMSで実行するビジネスルールの管理，実行を一元的に行う特許庁システム全体でただ一つの責務を持つため，階層化の対象外とする。
- 層間の依存関係を統制するためアクセスパスの方向は同位層か下位層へ向かう向きとする。

3.1.1.2.3 層内のシステム構成要素の関係

層内のシステム構成要素の関係については，各層の責務に従い以下に示す関係を持たせる。

- 特許庁業務を実現する上で，必要最小限のアクセスパスを設ける。
- 業務の流れをビジネスプロセスとして表すのはワークフロー層の責務であるため，業務アプリケーション層内において，業務アプリケーションの連鎖的な実行の流れによりビジネスプロセスが構成されないようにシステム構成要素間のアクセスを制限する。

3.1.1.3 多階層構造の詳細

目的:	「3.1.1.1 多階層構造を採用する目的」に示される要求を達成することを目的とする。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	サブシステムは「表 3.1-4 サブシステムの分類とシステム構成要素」に示された「サブシステムの分類」に分類すること。
規約2:	サブシステムの機能は,「表 3.1-3 システム構成要素の種類及び責務」に示された責務を担うシステム構成要素から実現すること。
規約3:	サブシステムの機能は,「表 3.1-4 サブシステムの分類とシステム構成要素」に示された, サブシステムの分類毎に定められたシステム構成要素群で構成すること。
規約4:	システム構成要素間のアクセスは,「3.1.1.3.1.3 (1)多階層構造の概要に基づくアクセスパス」に示すアクセスパスとすること。また,「表 3.1-7 アクセスパスの制限事項」に示すとおりアクセスパスを制限すること。
特例1:	「3.1.1.3.1.3 (4)アクセスパスの特例」に示すアクセスパスは, 下位層のシステム構成要素から上位層のシステム構成要素へのアクセスパスを許容する。
特例2:	「3.1.1.3.1.3 (5)システム構成要素の配置の特例及び特例となるシステム構成要素のアクセスパス」に示すとおり, 特例となるシステム構成要素の配置及びアクセスパスを許容する。
規約5:	個別データベースへのアクセスは,「3.1.1.3.1.3 (6) 個別データベースへのアクセスルール」に示すアクセスパスとすること。
特例1:	情報活用系サブシステムへデータ提供する場合は, 情報活用系サブシステムからの個別データベースへの直接アクセスを許容する。
規約6:	各システム構成要素のソフトウェア構成は,「表 3.1-12 ソフトウェア構成一覧」に示す論理ノード及び, ソフトウェア又はデータベースで構成すること。

目的:	サブシステムの処理方式及び処理方式を構成するシステム構成要素を定めることにより, サブシステム構造の定型化を促進し, 変更時の影響箇所を具体的に把握することを容易にする。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	サブシステムの処理方式は,「表 3.1-5 サブシステムの処理方式」に示す処理方式とすること。
規約2:	サブシステムの処理方式は,「表 3.1-6 処理方式とシステム構成要素」に示された, 処理方式毎に定められたシステム構成要素群で構成すること。

多階層構造の詳細を定めるにあたり，どの層にどのシステム構成要素が所属しているかといった関係や，どのシステム構成要素がどのシステム構成要素にアクセスするかといった関係を以下に示す多階層構造図によって定める。

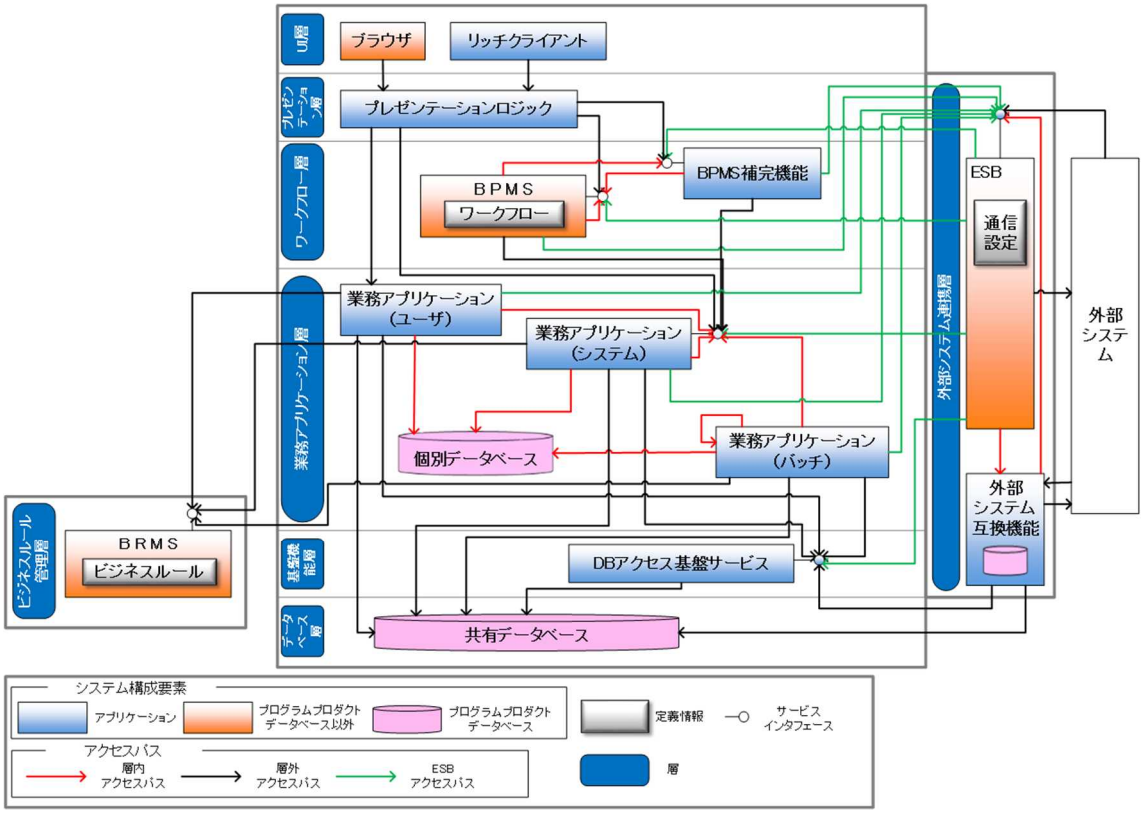
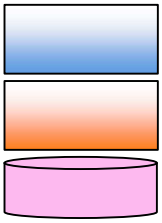


図 3.1-1 多階層構造図

表 3.1-2 多階層構造図における凡例の詳細

項番	要素	説明
1	層 	システム構成要素の所属する層を定義したもの。 例えば，UI層には「ブラウザ」と「リッチクライアント」の二つのシステム構成要素が属していることを示している。
2	システム構成要素 	システムが実現する機能を責務ごとの要素に分割したもの。 「表 3.1-3 システム構成要素の種類及び責務」に定義される責務をアプリケーション，プログラムプロダクト(データベース以外)，プログラムプロダクト(データベース)のいずれかによって実現することを示している[4]。 ● 四角形で青色のシステム構成要素は「表 3.1-3 システム構成要素の種類及び責務」の責務をアプリケーションにより実現する。 ● 四角形で橙色のシステム構成要素は「表 3.1-3 システム構成要素の種類及び責務」の責務をプログラムプロダクト(データベース以外)により実現する。 ● 円柱で桃色のシステム構成要素は「表 3.1-3 システム構成要素の種類及び責務」の責務をプログラムプロダクト(データベース)により実現する。 例えば，「BPMS」はワークフローの制御による業務アプリケーションの連携という責務をプログラムプロダクトによって実現しなければならない。 なお，システム構成要素間の関係は，後述するアクセスパスに従うこと。
3	定義情報	プログラムプロダクトであるシステム構成要素の責務のうち，業務要件や機能要件を実現するために設計・開発する定義情報を示すもの。 この定義情報はプログラムプロダクト上で動作させることにより，システム

[4] プログラムプロダクトを利用することで，アプリケーションの作り込みを極力避け，製品の設定情報の変更等の対応により保守性・移植性・開発効率性を高める効果を狙う。

項番	要素	説明
		構成要素の責務を担う。本要素は，プログラムプロダクトであっても設計に関与するステークホルダ（設計・開発ベンダ等）において開発が必要となる要素であり，アーキテクチャ上のルールを説明するための要素ではなく，補足的に書き加えたものである。
4	サービスインタフェース 	システム構成要素がサービスインタフェースを提供することを示すもの。 ルールの詳細は，「3.2.1.1 内部インタフェースのルール」を参照のこと。
5	アクセスパス[5] 	システム構成要素間のアクセスの向きを矢印で示すもの。 アクセスパスが存在するシステム構成要素は，矢印の向きにのみアクセスが可能であり，したがってアクセス元はアクセス先のシステム構成要素の変更による直接的な影響を受ける可能性があり，その逆の向きにはないことを意味する。また，アクセスパスが存在しないシステム構成要素間は直接のアクセスを持たないためシステム構成要素の変更による直接的な影響を互いに受けないことを意味する。 例えば，「プレゼンテーションロジック」と「業務アプリケーション（ユーザ）」，「個別データベース」において，「プレゼンテーションロジック」から「業務アプリケーション（ユーザ）」へのアクセスは可能だが，「業務アプリケーション（ユーザ）」から「プレゼンテーションロジック」へはアクセスできない。また，「プレゼンテーションロジック」と「個別データベース」は矢印でつながっていないため，直接アクセスはできない。 また，アクセスパスがシステム構成要素のサービスインタフェースを指している場合，アクセス元はサービスインタフェースを介してアクセスをすること。ルールの詳細は，「3.2.1.1 内部インタフェースのルール」を参照のこと。

なお，本書に示す同様の多階層構造図において，各要素の凡例は「表 3.1-2 多階層構造図における凡例の詳細」に従う。

[5] 見易さのため，矢印を層内のシステム要素同士のアクセス（赤），層間のシステム構成要素同士のアクセス（黒），層間のシステム構成要素同士のアクセスのうち一方のシステム構成要素がESBの場合のアクセス（緑）で色分けしている。

3.1.1.3.1 システム構成要素の種類及び責務

特許庁システム全体として必要とする機能を抽出するとともに「3.1.1.2 多階層構造の概要」に基づきシステム構成要素を整理し，システム構成要素の種類及び各システム構成要素に割り当てられた責務として下表に示す。表中のシステム構成要素の責務がサブシステムを実現する際に不要である場合，該当のシステム構成要素はサブシステムから除外すること。例えば，あるサブシステムのUIがブラウザのみで実現可能ならば，リッチクライアントをサブシステムに含めない。逆に，表中のシステム構成要素以外に新たにシステム構成要素を定義し追加してはならない。つまり，この表はサブシステムを実現する上で最大でこれだけのシステム構成要素から構成されるということを示している。

なお，システム構成要素の責務の詳細は「3.3 多階層構造の層毎に遵守すべきルール」を参照のこと。

表 3.1-3 システム構成要素の種類及び責務

項番	層	システム構成要素	責務
1	UI層	ブラウザ	必要な情報（HTML，CSS，画像ファイル等）を都度サーバから取得・解釈する，ユーザへの画面機能の提供
2		リッチクライアント	必要な情報（リッチクライアント固有の形式）を全て事前に配備・使用する，ユーザへの画面機能の提供
3	プレゼンテーション層	プレゼンテーションロジック	ビジネスロジックを含まず，UI層で用いられるデータ構造又はデータ形態と下位層のシステム構成要素で用いられるデータ構造又はデータ形態を相互変換する機能の提供
4	ワークフロー層	BPMS	ワークフローの制御による業務アプリケーションの連携
5		BPMS補完機能	ワークフローの再開箇所の判定等，BPMSにおけるワークフロー制御の補完
6	業務アプリケーション層	業務アプリケーション（ユーザ）	ユーザからの入力情報に基づく業務処理又はユーザが必要とする情報の取得
7		業務アプリケーション（システム）	以下に示す責務のいずれか ● BPMSのサービスタスクに対応し業務として意味のある一連の処理の実行 ● 他のシステム構成要素で必要とするデータの取得又は更新
8		業務アプリケーション（バッチ）	BPMSのサービスタスクに対応せず所定のタイミング（期間，時間又はディレード処理要求の発生等）で処理の実行
9		個別データベース	サブシステム固有の業務データの管理
10	基盤機能層	DBアクセス基盤サービス	共有データベースにアクセスするためのサービスの提供
11	データベース層	共有データベース	システム全体で利用する共有データ

項番	層	システム構成要素	責務
			の管理
12	外部システム連携層	ESB	ギャップの吸収における内部システムと外部システムのアクセスの一元化
13		外部システム互換機能	ギャップのうちESBのみでは吸収できない部分の補完
14	ビジネスルール管理層	BRMS	ビジネスルールの管理, 実行

3.1.1.3.1.1 サブシステムの分類に基づくシステム構成要素

多階層構造の各層に配置されたシステム構成要素は，以下のとおり2グループに分類できる。

① 業務又は事件毎に構築されるため定型化が必要なシステム構成要素のグループ

本グループに属するシステム構成要素は「表 3.1-3」の項番2～9並びに10及び11である。当該グループに含まれるシステム構成要素は各々の依存関係が極めて強いことから，初期構築時に一体として設計を行う必要があり，1サブシステムの範囲として含める。

当該サブシステムは，階層化した複数の層から構成され，業務又は事件毎に構築される。各サブシステムのアーキテクチャは定型化されていることから，階層定型化サブシステムと呼ぶ。

特に，項番2～9のシステム構成要素からなるサブシステムを，業務遂行・共通リソース系サブシステムと呼び，項番10及び11のシステム構成要素からなるサブシステムを，共有データベース管理サブシステムと呼ぶ。

② 共通的に利用される単品扱いのシステム構成要素のグループ

本グループに属するシステム構成要素は「表 3.1-3」の項番12以降である。当該システム構成要素は，責務が類似し依存関係の強いものもあれば，責務が全く異なり依存関係の薄いものもある。類似する責務のシステム構成要素を各々まとめてサブシステムの範囲とする。当該サブシステムは，インタフェースを定型化するものの，それ以外については個別にアーキテクチャを定める性質を有することから，インタフェース定型化サブシステムと呼ぶ[6]。

サブシステムの分類とシステム構成要素の関係を下表に示す。なお，この表はサブシステムを実現する上で最大でこれだけのシステム構成要素から構成されるということを示している。また，システム構成要素のうちブラウザについては業務用PCにインストールされるプログラムプロダクトとなるため，どのサブシステムの範囲にも属さないものとする。

表 3.1-4 サブシステムの分類とシステム構成要素

項番	サブシステムの分類		サブシステム	システム構成要素
1	階層定型化サブシステム	業務遂行・共通リソース系サブシステム[7]	方式審査（特許・実用），分類付与，実体審査 等	リッチクライアント，プレゼンテーションロジック，BPMS，BPMS補完機能，業務アプリケーション（ユーザ），業務アプリケーション（システム），業務アプリケーション（バッチ），個別データベース
		共有データベース管理サブシステム	共有データベース管理（特許・実用），共有データベース管理（審判） 等	DBアクセス基盤サービス，共有データベース
2	インタフェース定型化サブシステム		ビジネスルール管理	BRMS
			外部システム連携	ESB，外部システム互換機能
			情報活用系，リソース管理，運用監視，対外連携，紙出力	－

[6] 特に，システム構成要素としてBRMSを含むものをビジネスルール管理サブシステムと呼び，システム構成要素としてESB及び外部システム互換機能を含むものを外部システム連携サブシステムと呼ぶ。

[7] 業務的な責務の視点から，特許庁における個々の業務の遂行や，個々の業務に共通に現れる業務（申請人登録，料金納付等）の遂行を責務とするシステム構成要素のグループのこと。

多階層構造図におけるサブシステムの範囲を下図に示す。

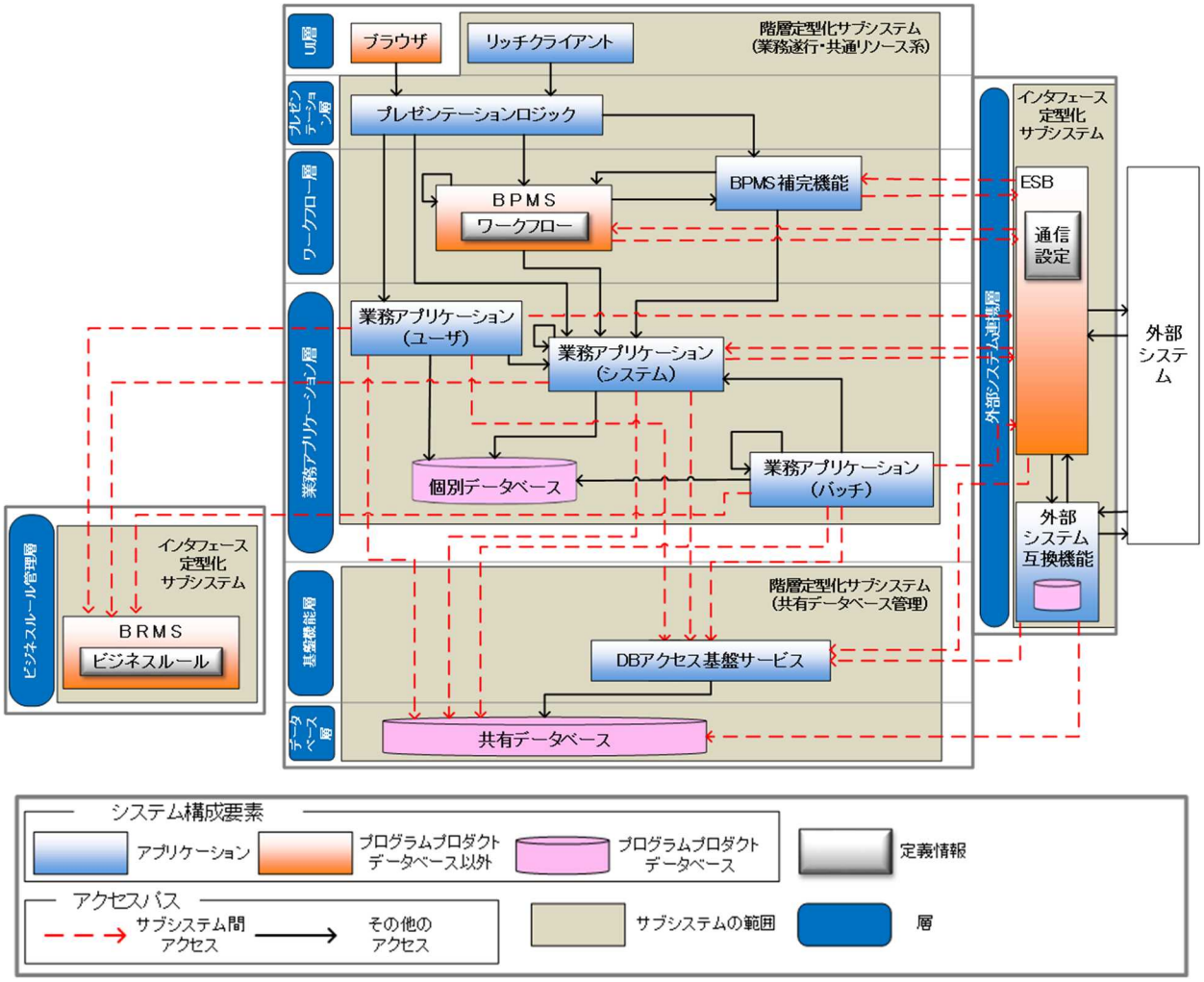


図 3.1-2 サブシステムの範囲[8]

[8] アクセスパスとサブシステムの範囲以外の凡例の詳細は,「表 3.1-2 多階層構造図における凡例の詳細」を参照のこと。

3.1.1.3.1.2 処理方式に基づくシステム構成要素

システムの構造を定型化するため，代表的な処理の処理方式を分類し定める。処理方式の分類を下表に示す。

各処理方式におけるAPレイヤ・コンポーネント定義は，「3.1.2.1」及び「3.1.2.2」以降の当該章節を参照のこと。各処理方式における動的な振る舞いは，「3.1.2 システムの動的な振る舞い」を参照のこと。

表 3.1-5 サブシステムの処理方式

項番	処理方式	分類	説明
1	オンライン処理方式	画面系	ユーザ操作により，画面からネットワークを介してサーバに処理要求を行い，応答を返却する方式。
2		サービス系	BPMS等からの処理要求に対し，サービスとして応答を返却する方式。
3	バッチ処理方式	複数件一括処理	多量のデータを取りまとめ，一定期間あるいはデータ量毎に一度にまとめて処理する方式。 一括処理単位は排他的に処理を行い，一括処理単位でリランを行う。リランによる再実行可能なケースで採用される。
4		単件一括処理	単件の処理を一度にまとめて処理する方式。 単件ごとに処理が完結するケースであり，処理済み案件はリランしない。単件処理は要件により逐次又は並行で実行する。
5	ディレードバッチ処理方式	画面系	ユーザ操作により，画面からサーバに処理要求を行い，要求受領の旨の応答を即時に返却した後，要求された処理を行う方式。
6	帳票出力方式 ※本書では処理方式に関する概要を説明するにとどめる。理由は「3.1.2.5 帳票出力方式」を参照のこと。	クライアント帳票出力	サーバで作成した帳票ファイルをクライアントにダウンロードする方式。
7		サーバ帳票出力（同期）	サーバで帳票ファイルの作成と，帳票ライブラリへの印刷要求を同期処理で行う方式。 印刷は，業務要件によってフロアプリンタ又はサーバプリンタに印刷可能とする。
8		サーバ帳票出力（非同期）	サーバで帳票ファイルの作成と，帳票ライブラリへの印刷要求を非同期処理で行う方式。 印刷は，業務要件によってフロアプリンタ又はサーバプリンタに印刷可能とする。
9	サブシステム間連携処理方式	BPMSを介したサブシステム間連携	業務フローとしてサブシステム間の連携を行う場合は，その連携ルートをBPMS上のワークフローにて記述し，BPMSを介して他サブシステムのBPMSや業務アプリケーション（システム）と連携を行う方式。
10		BPMSとジョブ管理システム間の連携	ToBeアーキテクチャでは，基本的にBPMSによって単件オンライン化されるが，一部の業務では業務アプリケーション（バッチ）により一括で処理を行うものも存在する。それらの処理を連携する処理方式。 オンライン処理方式（サービス系）とバッチ処理方式を組み合わせて，データベースやファイルでデータの授受を行い，スケジューラ起動等の契機により連携処理を行う。
11		バッチ処理方式のサブシステム間連携	サブシステム間にて，バッチ処理の連動やデータの授受を行う方式。 共有データベースやFTP(S)によるデータ授受を行い，ジョブ管理システムを介して連携を行う。

項番	処理方式	分類	説明
12		外部システム連携層を介した外部システム連携	外部システムとの連携には，それぞれの外部システムに固有のインタフェースやプロトコルが存在するため，それらのギャップを一元的に吸収する外部システム連携層を介してのみ行うものとする。その際の処理方式。 内部システムから外部システム連携を行う際，外部システム連携層が提供する内部インタフェースと同等のアクセス方法を用いて処理を実施する。

オンライン処理方式，バッチ処理方式及びディレードバッチ処理方式の処理方式の分類の範囲を，下図に示す。

なお，帳票出力方式及びサブシステム間連携処理方式は，オンライン処理方式やバッチ処理方式と組み合わせて実行される。

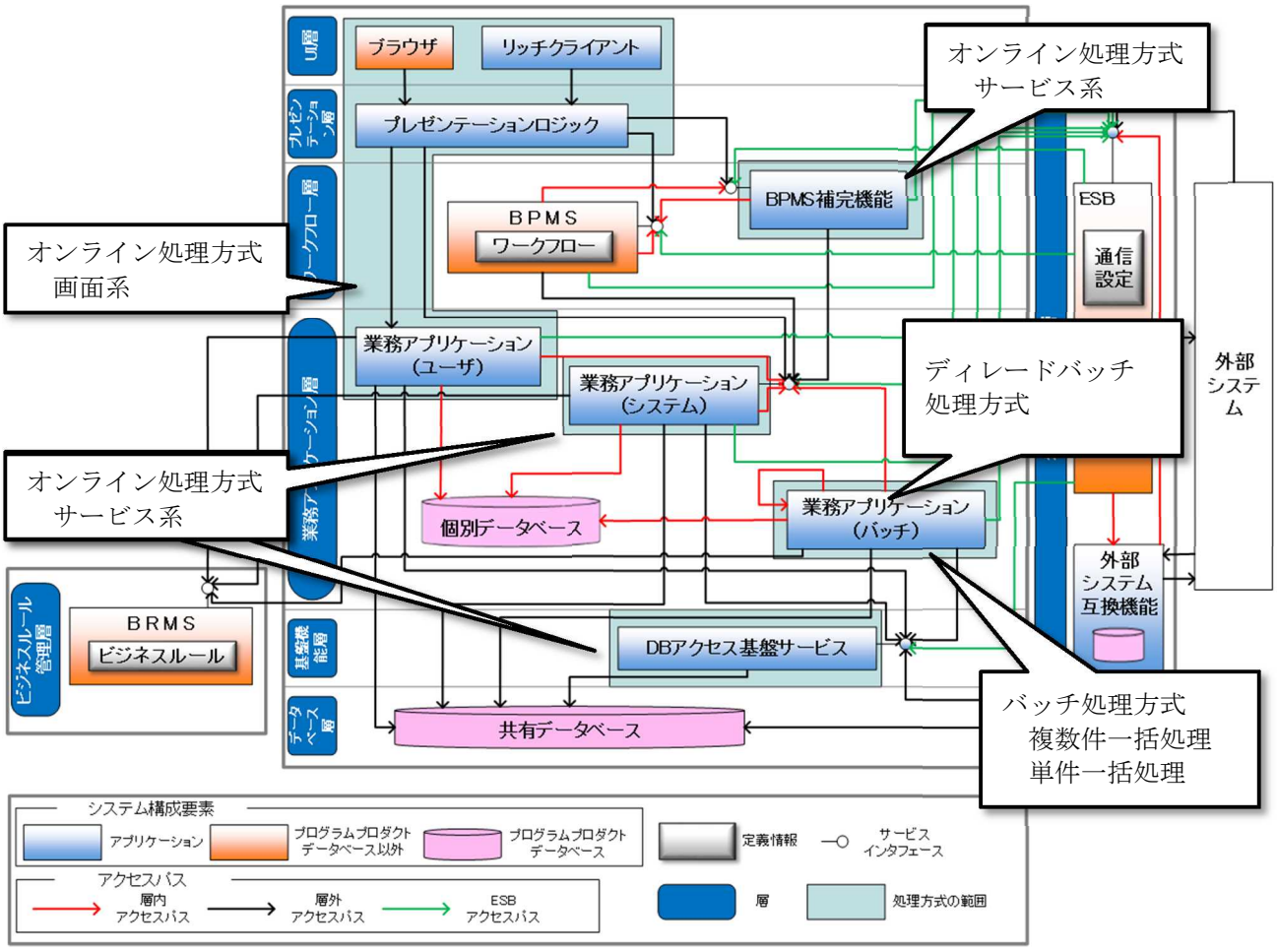


図 3.1-3 処理方式の分類の範囲[9]

[9] 処理方式の範囲以外の凡例の詳細は，「表 3.1-2 多階層構造図における凡例の詳細」を参照のこと。

処理方式毎のシステム構成要素について，以下の表にまとめる。

表 3.1-6 処理方式とシステム構成要素

項番	処理方式	分類	システム構成要素
1	オンライン処理方式	画面系	ブラウザ，リッチクライアント，プレゼンテーションロジック，業務アプリケーション（ユーザ）
2		サービス系	業務アプリケーション（システム），DBアクセス基盤サービス，BPMS補完機能
3	バッチ処理方式	複数件一括処理	業務アプリケーション（バッチ）
4		単件一括処理	
5	ディレードバッチ処理方式	画面系	
6	サブシステム間連携処理方式	BPMSを介したサブシステム間連携	業務アプリケーション（システム），DBアクセス基盤サービス，BPMS補完機能
7		BPMSとジョブ管理システム間の連携	業務アプリケーション（システム），業務アプリケーション（バッチ），DBアクセス基盤サービス
8		バッチ処理方式のサブシステム間連携	業務アプリケーション（バッチ），DBアクセス基盤サービス
9		外部システム連携層を介した外部システム連携	外部システム互換機能，業務アプリケーション（ユーザ），業務アプリケーション（システム），業務アプリケーション（バッチ），DBアクセス基盤サービス，BPMS補完機能

3.1.1.3.1.3 システム構成要素間のアクセスパス

(1)　多階層構造の概要に基づくアクセスパス

「3.1.1.2 多階層構造の概要」に基づきシステム構成要素間のアクセスパスを整理すると以下のとおりとなる。

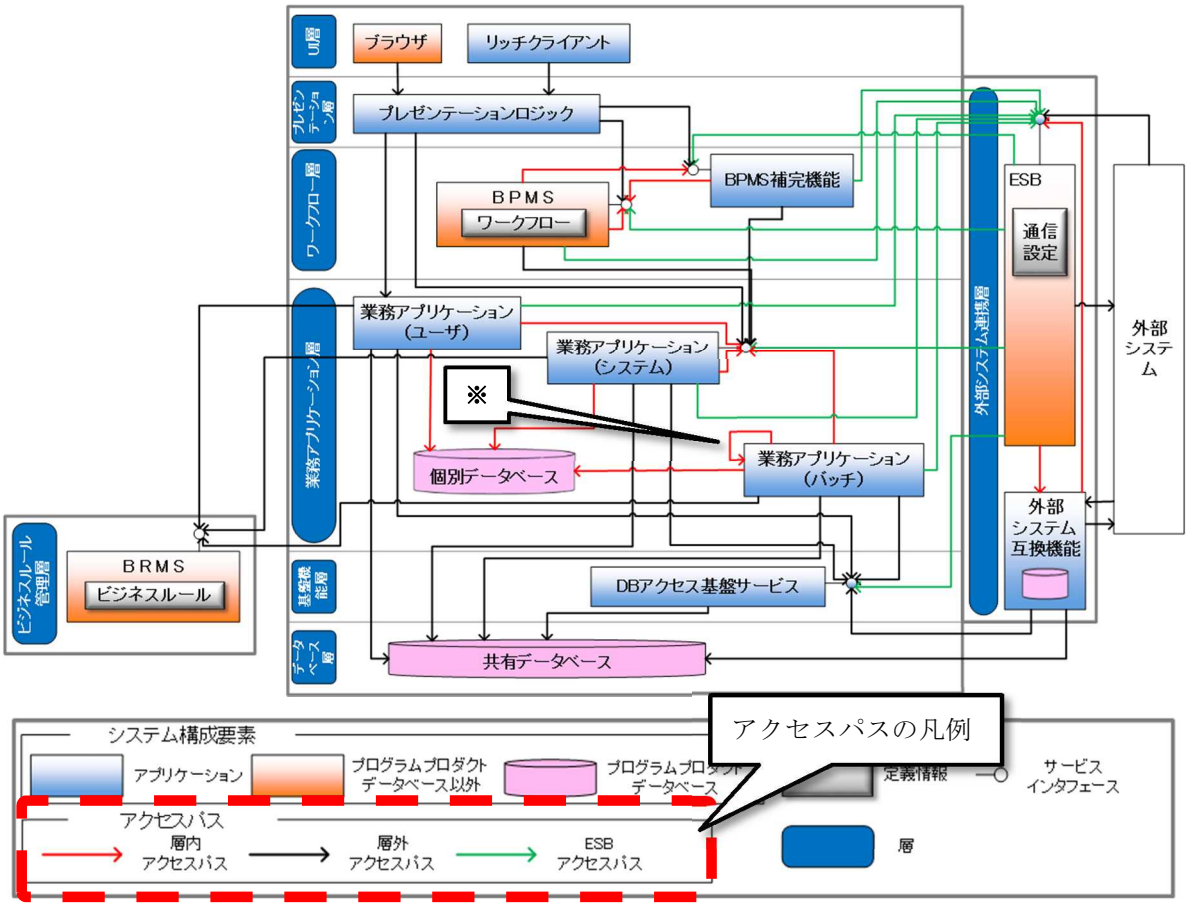


図 3.1-4 アクセスパス[10]

なお，図の※で示すアクセスパス（業務アプリケーション（バッチ）から業務アプリケーション（バッチ））は，同一階層内パスではあるが，自サブシステムから他サブシステムへの連携を表す。詳細は，「3.1.2.6.1.3 バッチ処理方式のサブシステム間連携」を参照のこと。

[10] 凡例の詳細は，「表 3.1-2 多階層構造図における凡例の詳細」を参照のこと。

(2)　業務アプリケーション(システム)のアクセスパスの制限事項

業務アプリケーション(システム)へのアクセス(下図の太線)は，アプリケーション同士の複雑な連携を排除しお互いを疎の関係とする目的から，提供するサービスに応じてアクセスを許容するシステム構成要素を制限する。

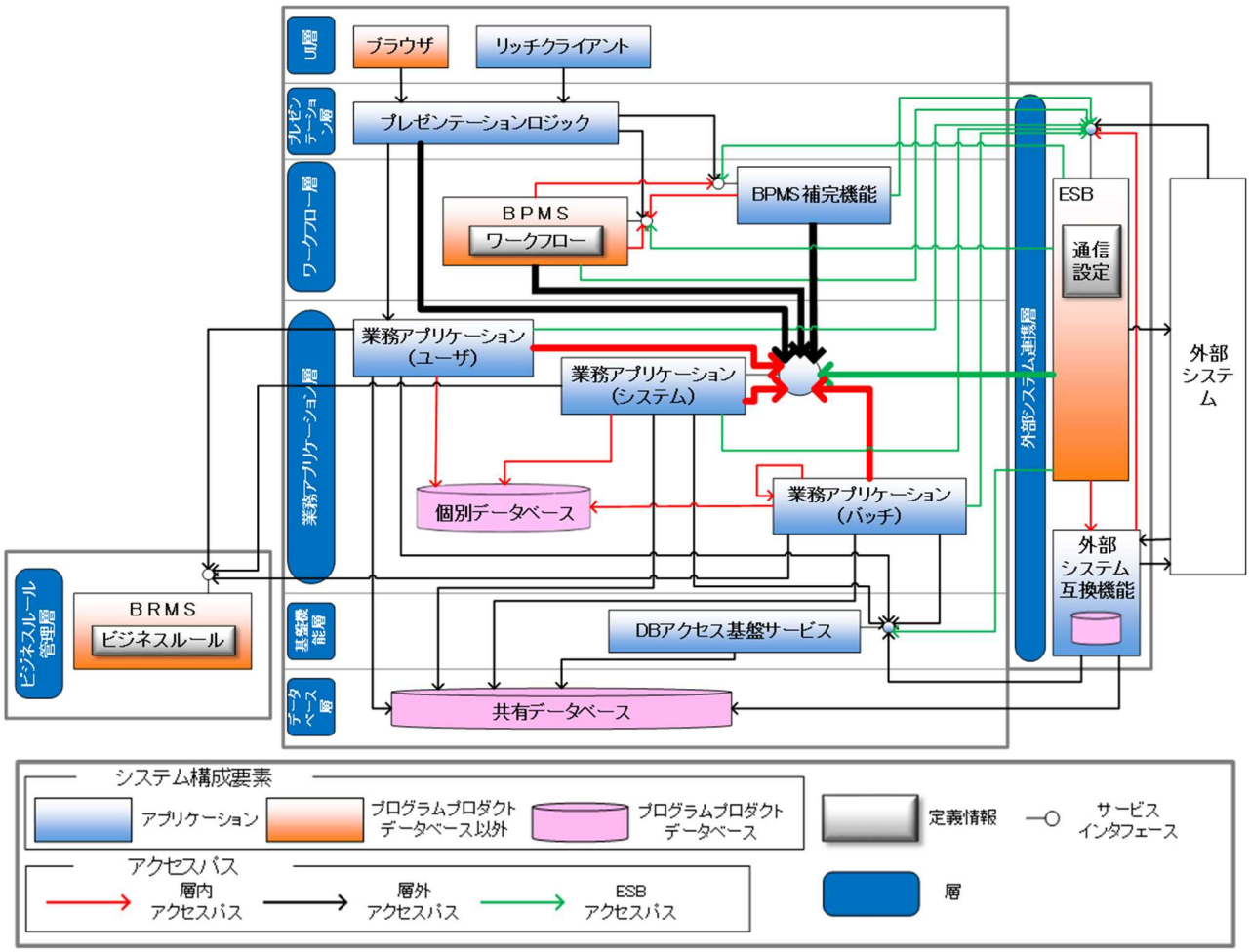


図 3.1-5 制限事項を持つアクセスパス[11]

図の太線で示した個々のアクセスパスについて制限事項を示したものが下表となる。表の「業務アプリケーション(システム)が提供するサービスの概要」列に記載したサービスに限り，表の「サービス利用側のシステム構成要素」列に記載しているシステム構成要素から業務アプリケーション(システム)へのアクセスを許可する。下表に示すアクセスパスの制限事項に従うこと。

表 3.1-7 アクセスパスの制限事項

項番	業務アプリケーション(システム)が提供するサービスの概要	サービス利用側のシステム構成要素
1	ビジネスプロセスのシステムタスクに対応するビジネスロジックを実行するためのサービス (業務単位のサービスインタフェースを提供)	・BPMS ・他サブシステムのプレゼンテーションロジック ※一つのアクティビティのみに相当する処理で，ビジネスプロセスとして意味を成さないため，BPMSを経由するオーバヘッドを考慮し，直接システムタスク相当の処理を呼び出す場合 ・ESB ※外部システムから業務単位のビジネスロジック実行要求を受け付ける場合

[11] 凡例の詳細は，「表 3.1-2 多階層構造図における凡例の詳細」を参照のこと。

項番	業務アプリケーション(システム)が提供するサービスの概要	サービス利用側のシステム構成要素
2	共通リソースデータの取得及び更新のためのサービス	・業務アプリケーション(ユーザ) ・業務アプリケーション(システム) ※項番1のサービスから利用する場合のみ ・業務アプリケーション(バッチ) ・他サブシステムのプレゼンテーションロジック ・ESB ※外部システムから共通リソースデータの操作要求を受け付ける場合
3	サブシステムが管理する個別データの取得及び更新のためのサービス	・他サブシステムのプレゼンテーションロジック ※更新においては，サブシステムをまたがった業務を同時に扱う必要がある場合のみ(例えば，国際調査と国際予備審査のセット起案)
4	BPMSサービスの補完処理に必要なデータ取得及び更新のためのサービス	・BPMS補完機能

「表 3.1-7 アクセスパスの制限事項」の業務アプリケーション(システム)へのアクセスのうち，項番1のシステムタスクに対応するビジネスロジックを実行するためのサービスと，項番2の共通リソースデータのアクセスのためのサービスについて，アクセスパスを図示したものを以下に示す。

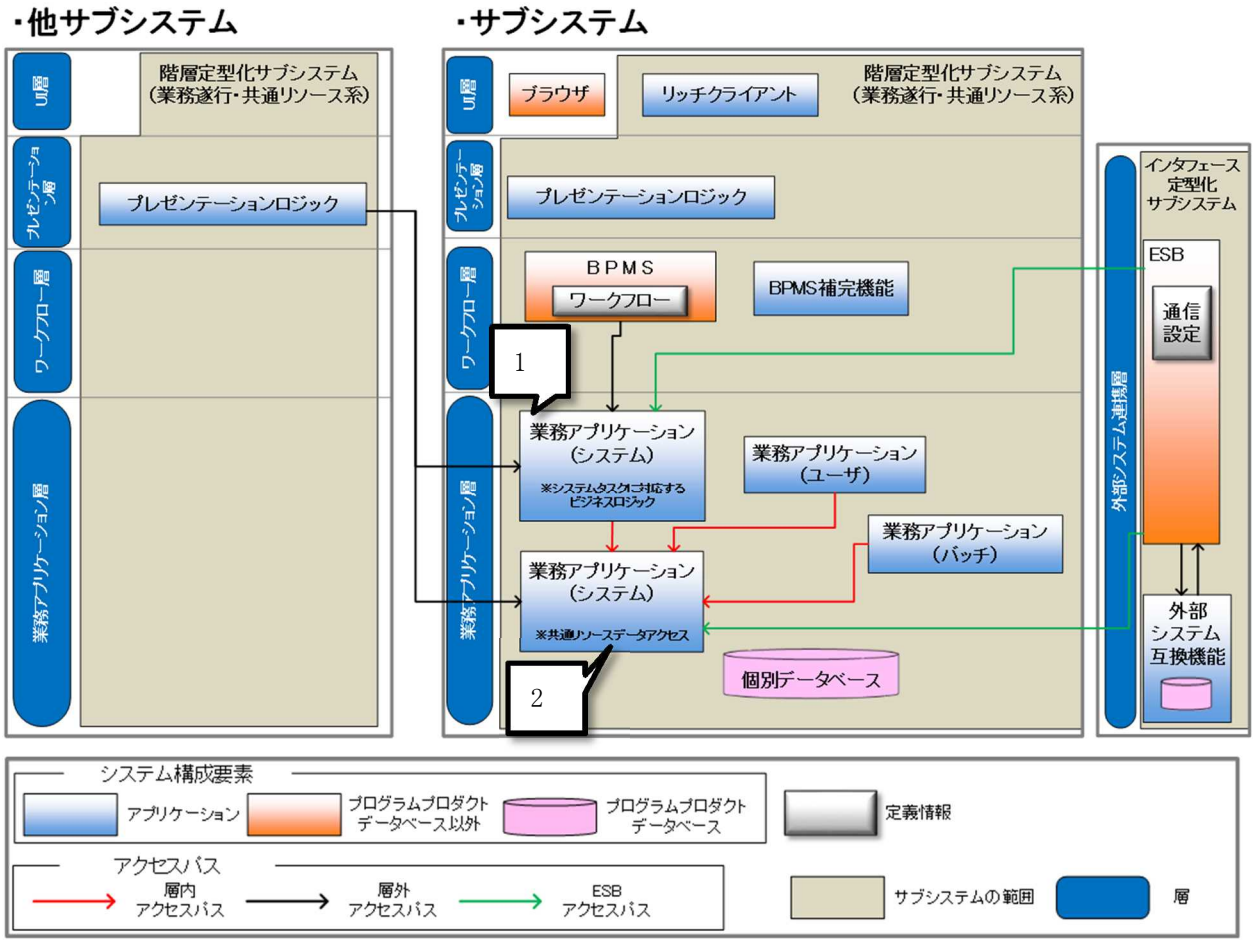


図 3.1-6 業務アプリケーション(システム)のアクセスパスの制限事項(項番1, 2)[12]

[12] 図中の番号は,「表 3.1-7 アクセスパスの制限事項」の項番を示す。サブシステムの範囲以外の凡例の詳細は,「表 3.1-2 多階層構造図における凡例の詳細」を参照のこと。

「表 3.1-7 アクセスパスの制限事項」の業務アプリケーション(システム)へのアクセスのうち，項番3のサブシステムが管理する個別データの取得及び更新のためのサービスについて，アクセスパスを図示したものを以下に示す。

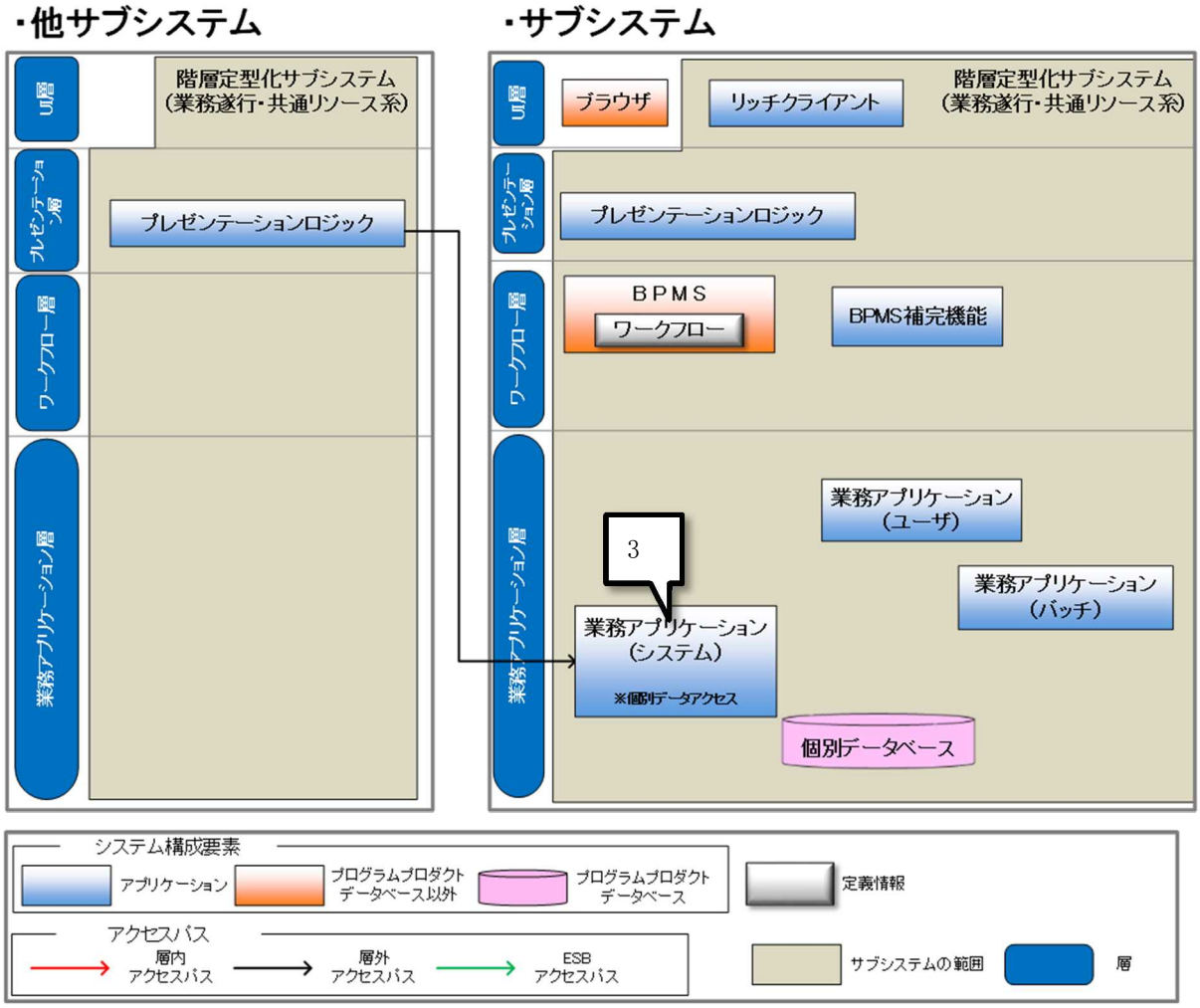


図 3.1-7 業務アプリケーション(システム)のアクセスパスの制限事項(項番3)[12]

「表 3.1-7 アクセスパスの制限事項」の業務アプリケーション(システム)へのアクセスのうち，項番4のBPMSサービスの補完処理に必要なデータ取得及び更新のためのサービスについて，アクセスパスを図示したものを以下に示す。

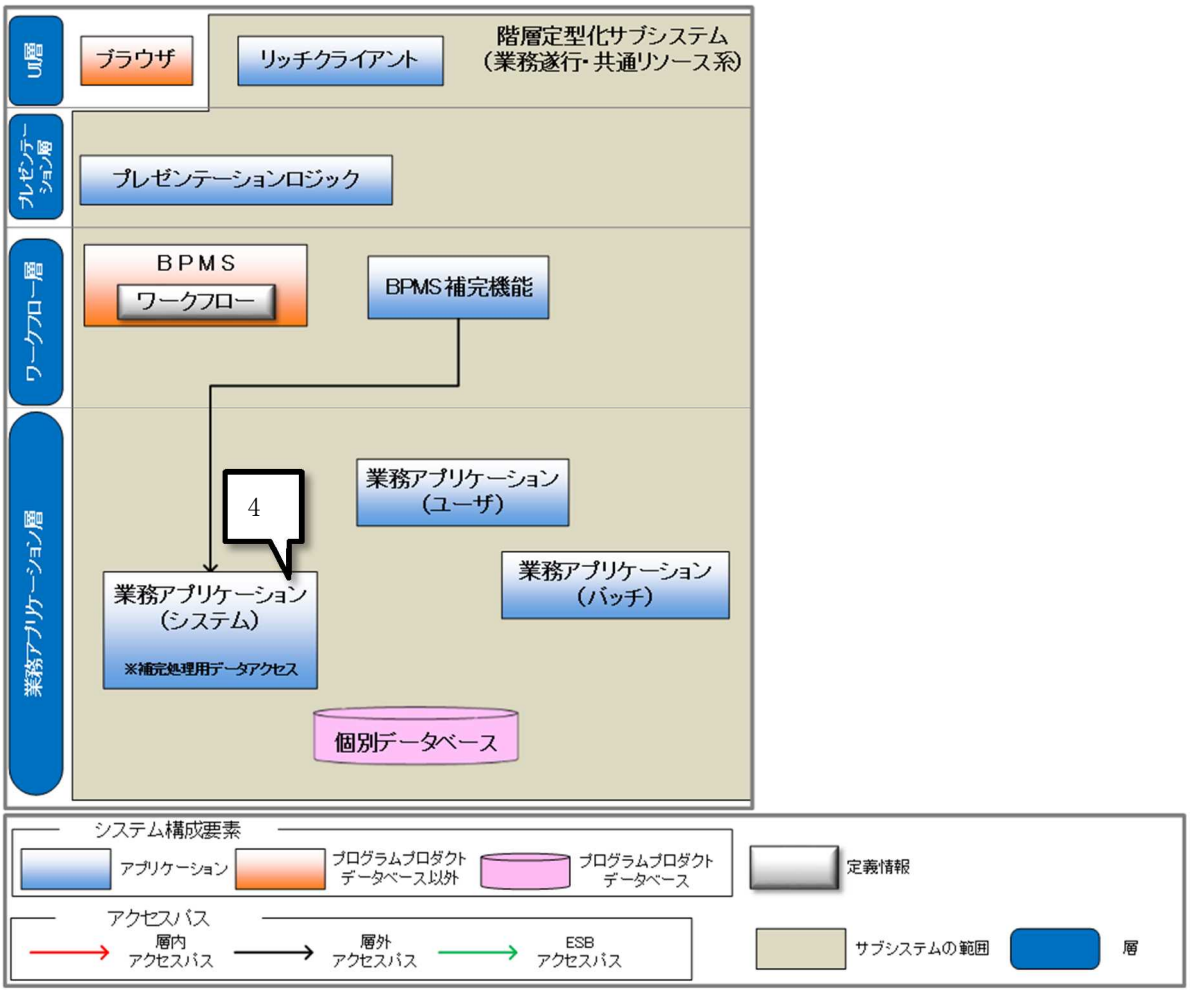


図 3.1-8 業務アプリケーション(システム)のアクセスパスの制限事項(項番4)[12]

(3)　アクセスパスの制限事項に関する補足

「表 3.1-7 アクセスパスの制限事項」で示したとおり，業務アプリケーション（システム）同士のアクセスは，項番1のシステムタスクに対応するビジネスロジックを実行するためのサービスから，項番2の共通リソースデータの取得及び更新のサービスへのアクセスのみに制限するルールとしている。

下図のように，項番1の業務アプリケーション同士や，項番2の業務アプリケーション同士のアクセスは，アプリケーションを疎の関係とする目的から禁止されていることに留意すること。

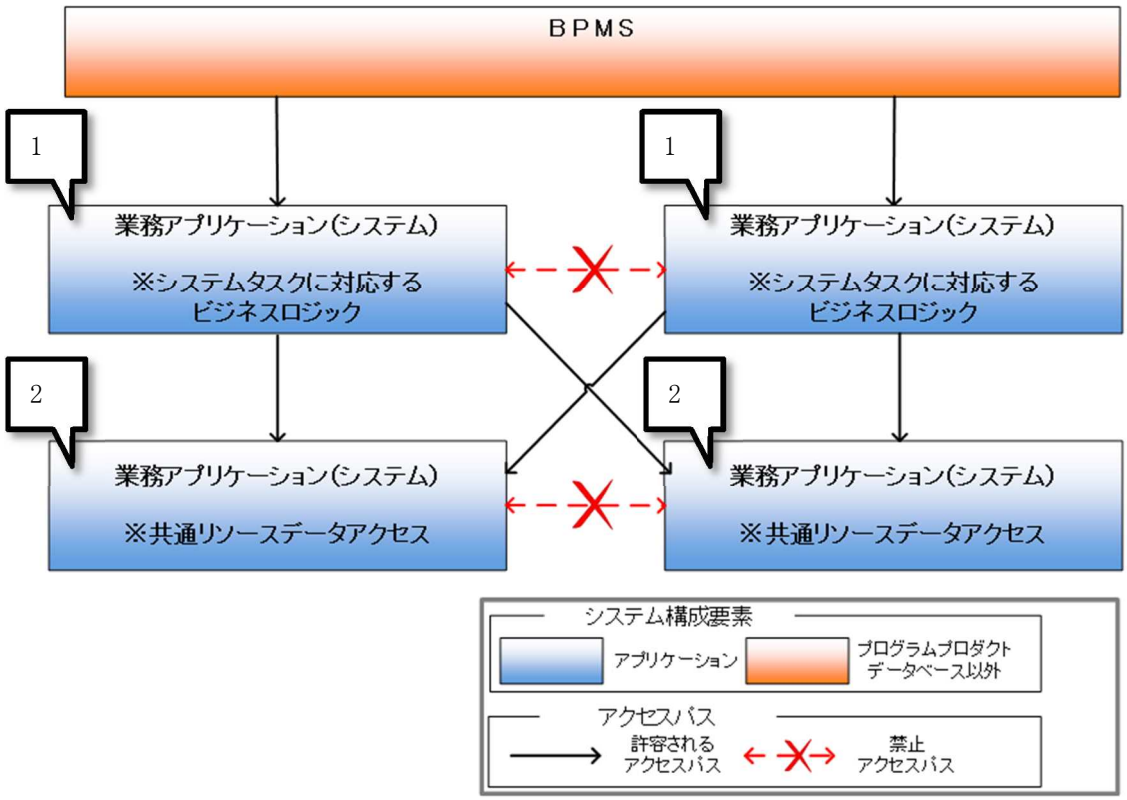


図 3.1-9 業務アプリケーション（システム）のアクセスパスの制限事項補足[13]

[13] 図中の番号は，「表 3.1-7 アクセスパスの制限事項」の項番を示す。

(4)　アクセスパスの特例

下表に示すアクセスパスについては，下位層のシステム構成要素から上位層のシステム構成要素へのアクセスパスとする。

表 3.1-8 アクセスパスの特例と許容されるアクセス条件

項番	アクセス元	アクセス先	許容されるアクセス条件
1	業務アプリケーション（システム）	BPMS BPMS補完機能[14]	● 業務アプリケーション（システム）が業務を遂行する上で，BPMSだけが保持する情報を必要とする際に，その情報を取得する目的でアクセスする場合（例えば，業務アプリケーション（システム）のビジネスロジックで，別のビジネスプロセスインスタンスのステータス情報に基づいた処理を行うためにステータス情報を取得する）。 ● 業務アプリケーション（システム）が業務を遂行する上で，BPMSだけが保持する情報の変更や状態の変更（ビジネスプロセスを進める等）を行う目的でアクセスする場合（例えば，人事異動のため，ビジネスプロセスインスタンスに割り当てられている担当者をオンラインで変更する）。
2	業務アプリケーション（バッチ）	BPMS BPMS補完機能[14]	● 業務アプリケーション（バッチ）の処理によるデータベースの更新等に伴い，データの整合性をとるためにBPMSが保持する情報の変更や状態の変更（ビジネスプロセスを進める等）を行う目的でアクセスする場合（例えば，人事異動のため，ビジネスプロセスインスタンスに割り当てられている担当者をバッチにて一括で変更する）。 ● 業務アプリケーション（バッチ）からビジネスプロセスを開始する目的でアクセスする場合。 ● 業務アプリケーション（バッチ）での業務的な期間の算出結果に基づき，イベントを通知する目的でアクセスする場合。

[14] BPMS補完機能は，将来的にBPMS製品によって実現できることを期待して補完するものであるため，アクセス元からは区別の必要がないようアクセスパスの特例もBPMSとBPMS補完機能の双方を許容する。

「表 3.1-8 アクセスパスの特例と許容されるアクセス条件」のアクセスパスを図で示すと，下図の赤の点線矢印になる。

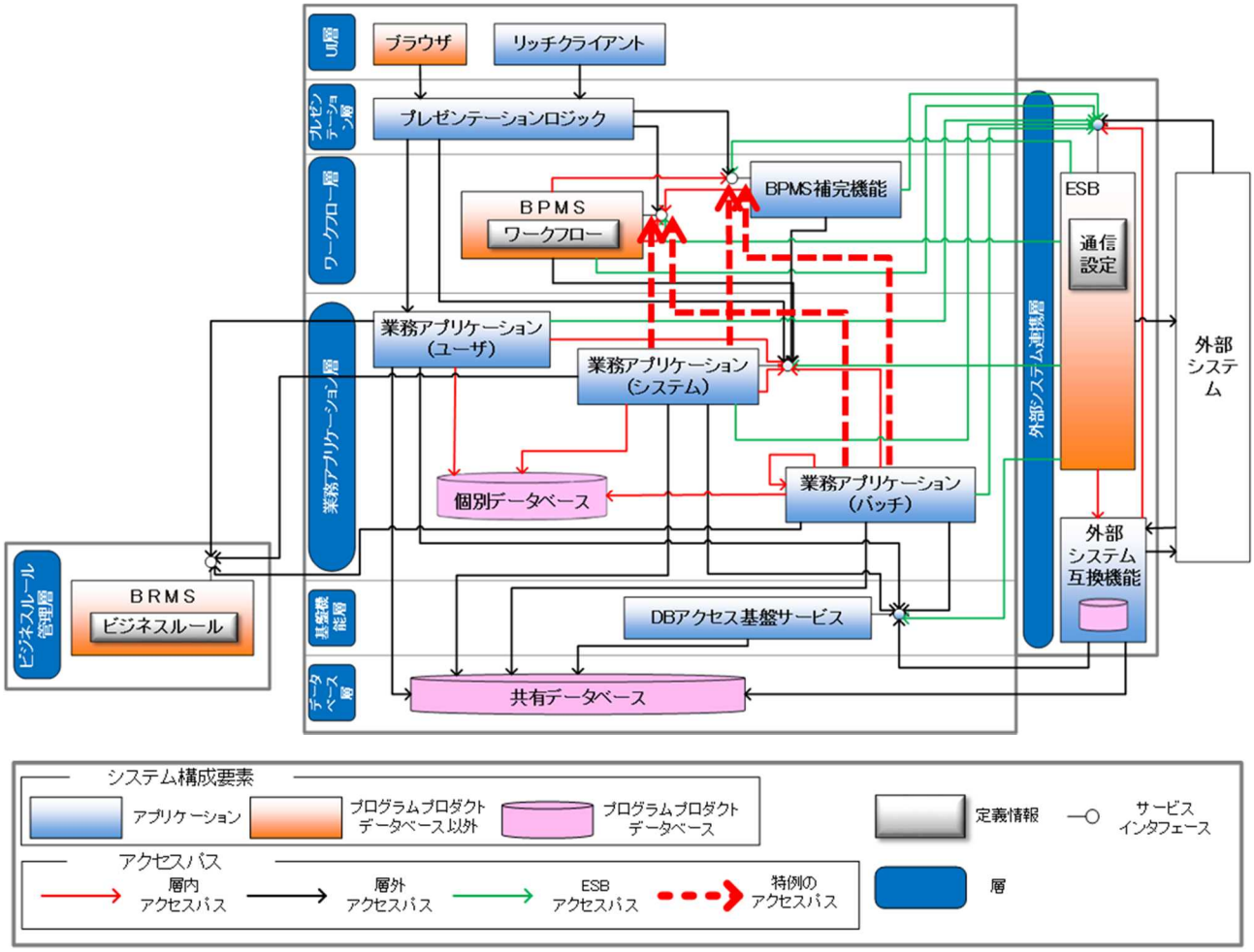


図 3.1-10 アクセスパスの特例[15, 16]

(5) システム構成要素の配置の特例及び特例となるシステム構成要素のアクセスパス

調査目的(事務処理を行うシステムでの事務処理の業務に含まれるもの以外)により，DBアクセス基盤サービスを利用して共有データベースの情報を参照する業務要件がある場合，後述する条件を満たす場合に限り，業務アプリケーションをUI層に配置することを特例として許容する。また，当該業務アプリケーションが，DBアクセス基盤サービスへの直接のアクセスパスを持つことをあわせて特例として許容する。

特例を許容する条件は以下のとおり。

- 当該業務アプリケーションからDBアクセス基盤サービスへのアクセスは参照のみとし，共有データの更新は行わないこと。
- この特例となるアクセス先のDBアクセス基盤サービス及び共有データベースは，調査目的のための専用のサーバ環境で稼動することとし，通常の業務処理を行う環境に影響を与えないようにすること。

[15] 特例のアクセスパス以外の凡例の詳細は，「表 3.1-2 多階層構造図における凡例の詳細」を参照のこと。

[16] BPMS補完機能は，将来的にBPMS製品によって実現できることを期待して補完するものであるため，アクセス元からは区別の必要がないようアクセスパスの特例もBPMSとBPMS補完機能の双方を許容する。

システム構成要素の配置の特例及び特例となるシステム構成要素のアクセスパスを下図に示す。赤の点線枠で囲った部分が特例として配置されるシステム構成要素，赤の点線矢印が特例のアクセスパスである。

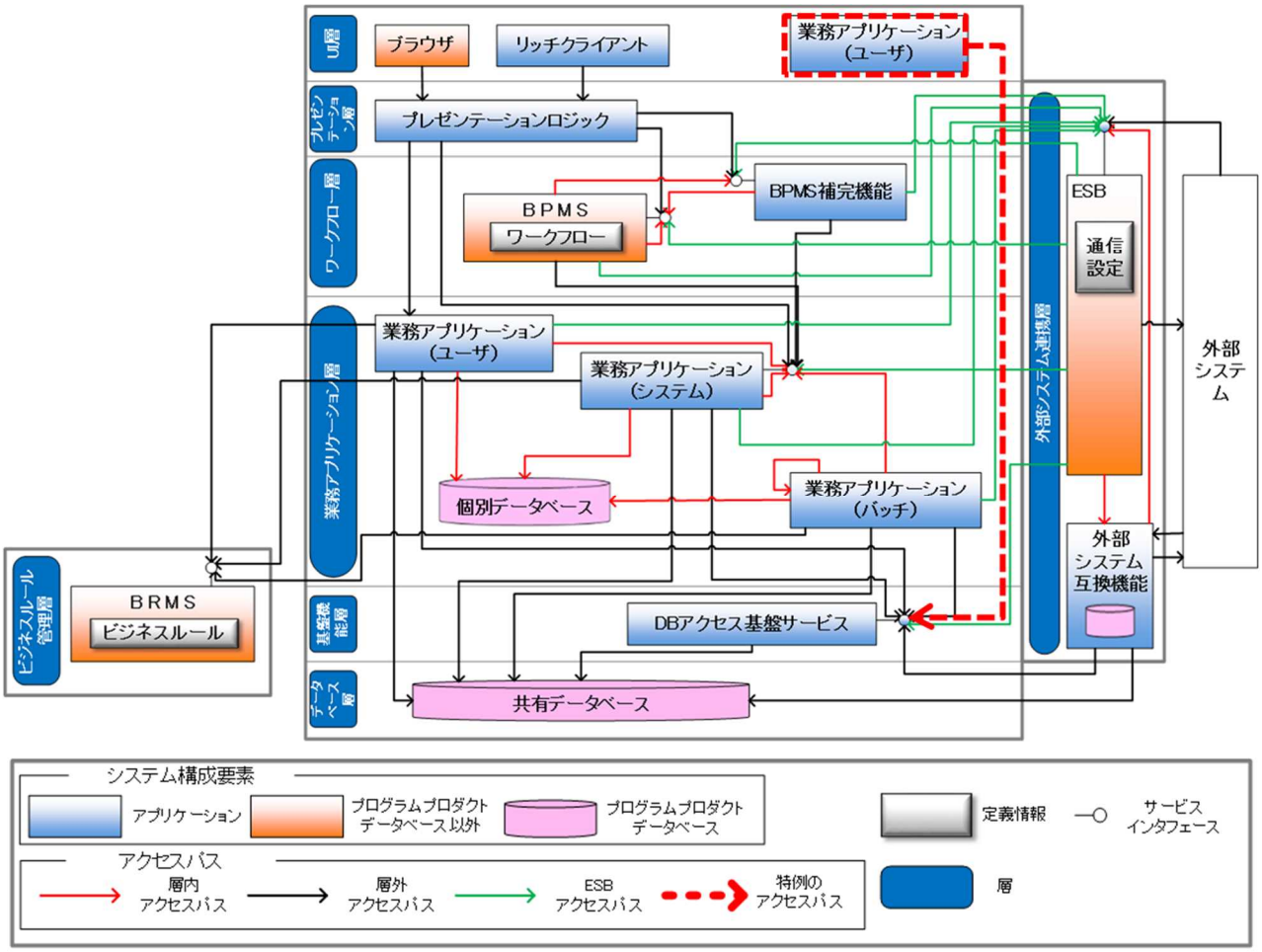


図 3.1-11 システム構成要素の配置の特例及び特例となるシステム構成要素のアクセスパス[17]

[17] 特例のアクセスパス以外の凡例の詳細は，「表 3.1-2 多階層構造図における凡例の詳細」を参照のこと。

(6)　個別データベースへのアクセスルール

個別データベースへのアクセスルールを以下に示す。

- 他サブシステムの個別データベースを直接アクセスすることを禁止し，他サブシステムが提供するサービスを呼び出すことで間接的にアクセスすること。
- 情報活用系サブシステムへデータ提供する場合は，情報活用系サブシステムから個別データベースへの直接アクセスを許容する。

個別データベースにアクセスするイメージを下図に示す。

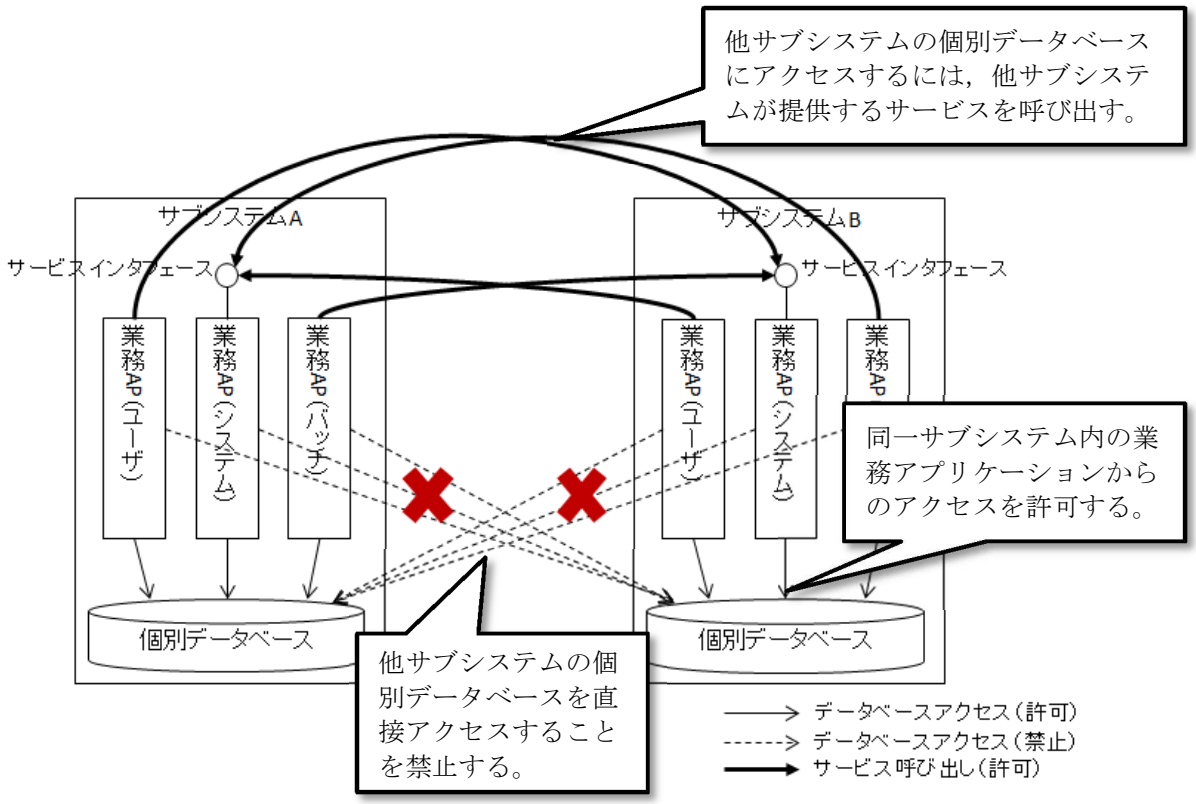


図 3.1-12 個別データベースへのアクセスイメージ

(7)　個別データベースへのアクセスルールの補足

個別データベースへのアクセスルールに従い，BRMS，BPMSから個別データベースへのアクセスは禁止する。

他サブシステムの個別データベースにアクセスするには，他サブシステムが提供するサービスを呼び出すものとする。ただし，業務アプリケーション（システム）同士の呼び出し関係は一定のルールに従う必要がある。詳細は，「3.1.1.3.1.3 (2)」を参照のこと。

(8)　アクセスパスのまとめ

前述の(1)から(7)で示したアクセスパスに関する方針について，連携元システムと連携先システムとの関係毎にまとめたものが，以下の表である。

なお，表中の「業務AP(システム・ビジネスロジック)」は「表 3.1-7 アクセスパスの制限事項」の項番1，「業務AP(システム・共通リソースアクセス)」は同項番2，「業務AP(システム・個別DBアクセス)」は同項番3，「業務AP(システム・BPMS補完処理用データアクセス)」は同項番4のサービスを示している。

- 自サブシステム内での連携

表 3.1-9 システム構成要素間のアクセスパス(自サブシステム内での連携)

				連携先(自サブシステム)																		
				階層定型化サブシステム(業務遂行・共通リソース系)													階層定型化サブシステム(共有データベース管理)		インタフェース定型化サブシステム		
				UI層		プレゼンテーション層	ワークフロー層		業務アプリケーション層								基盤機能層	データベース層	外部システム連携層		ビジネスルール管理層
				ブラウザ	リッチクライアント	プレゼンテーションロジック	BPMS	BPMS補完機能	業務AP(ユーザ)	業務AP(システム・ビジネスロジック)	業務AP(システム・共通リソースアクセス)	業務AP(システム・個別DBアクセス)	業務AP(システム・BPMS補完処理用データアクセス)	業務AP(バッチ)	個別データベース	DBアクセス基盤サービス	共有データベース	ESB	外部システム互換機能	BRMS
連携元(自サブシステム)	階層定型化サブシステム(業務遂行・共通リソース系)	UI層	ブラウザ			◎																
			リッチクライアント			◎																
		プレゼンテーション層	プレゼンテーションロジック				◎	◎	◎													
		ワークフロー層	BPMS				◎	◎		○												
			BPMS補完機能				◎						○									
		業務アプリケーション層	業務AP(ユーザ)								○				◎							
			業務AP(システム・ビジネスロジック)				△	△			○				◎							
			業務AP(システム・共通リソースアクセス)												◎							
			業務AP(システム・個別DBアクセス)												◎							
			業務AP(システム・BPMS補完処理用データアクセス)												○							
			業務AP(バッチ)				△	△			○				◎							
			個別データベース																			
	階層定型化サブシステム(共有データベース管理)	基盤機能層	DBアクセス基盤サービス															◎				
		データベース層	共有データベース																			
	インタフェース定型化サブシステム	外部システム連携層	ESB							○	○									◎		
			外部システム互換機能															◎				
		ビジネスルール管理層	BRMS																			

＜凡例＞
「◎」：アクセス可(制限なし)
「○」：アクセス可(制限あり)
「△」：アクセスパスの特例
「＼」(斜線)：アクセス不可

- 他サブシステムとの連携

表 3.1-10 システム構成要素間のアクセスパス(他サブシステムとの連携)

				連携先(他サブシステム)																	
				階層定型化サブシステム(業務遂行・共通リソース系)													階層定型化サブシステム(共有データベース管理)		インタフェース定型化サブシステム		
				UI層		プレゼンテーション層	ワークフロー層		業務アプリケーション層								基盤機能層	データベース層	外部システム連携層		ビジネスルール管理層
				ブラウザ	リッチクライアント	プレゼンテーションロジック	BPMS	BPMS補完機能	業務AP(ユーザ)	業務AP(システム・ビジネスロジック)	業務AP(システム・共通リソースアクセス)	業務AP(システム・個別DBアクセス)	業務AP(システム・BPMS補完処理用データアクセス)	業務AP(バッチ)	個別データベース	DBアクセス基盤サービス	共有データベース	ESB	外部システム互換機能	BRMS
連携元(自サブシステム)	階層定型化サブシステム(業務遂行・共通リソース系)	UI層	ブラウザ																		
			リッチクライアント																		
		プレゼンテーション層	プレゼンテーションロジック				◎	◎		○	○	○									
		ワークフロー層	BPMS				◎	◎		○									◎		
			BPMS補完機能																◎		
		業務アプリケーション層	業務AP(ユーザ)								○					◎	◎	◎		◎
			業務AP(システム・ビジネスロジック)								○					◎	◎	◎		◎
			業務AP(システム・共通リソースアクセス)																		
			業務AP(システム・個別DBアクセス)																		
			業務AP(システム・BPMS補完処理用データアクセス)																		
			業務AP(バッチ)								○			◎		◎	◎	◎		◎
			個別データベース																		
	階層定型化サブシステム(共有データベース管理)	基盤機能層	DBアクセス基盤サービス																		
		データベース層	共有データベース																		
	インタフェース定型化サブシステム	外部システム連携層	ESB				◎	◎		○	○					◎					
			外部システム互換機能													◎	◎				
		ビジネスルール管理層	BRMS																		

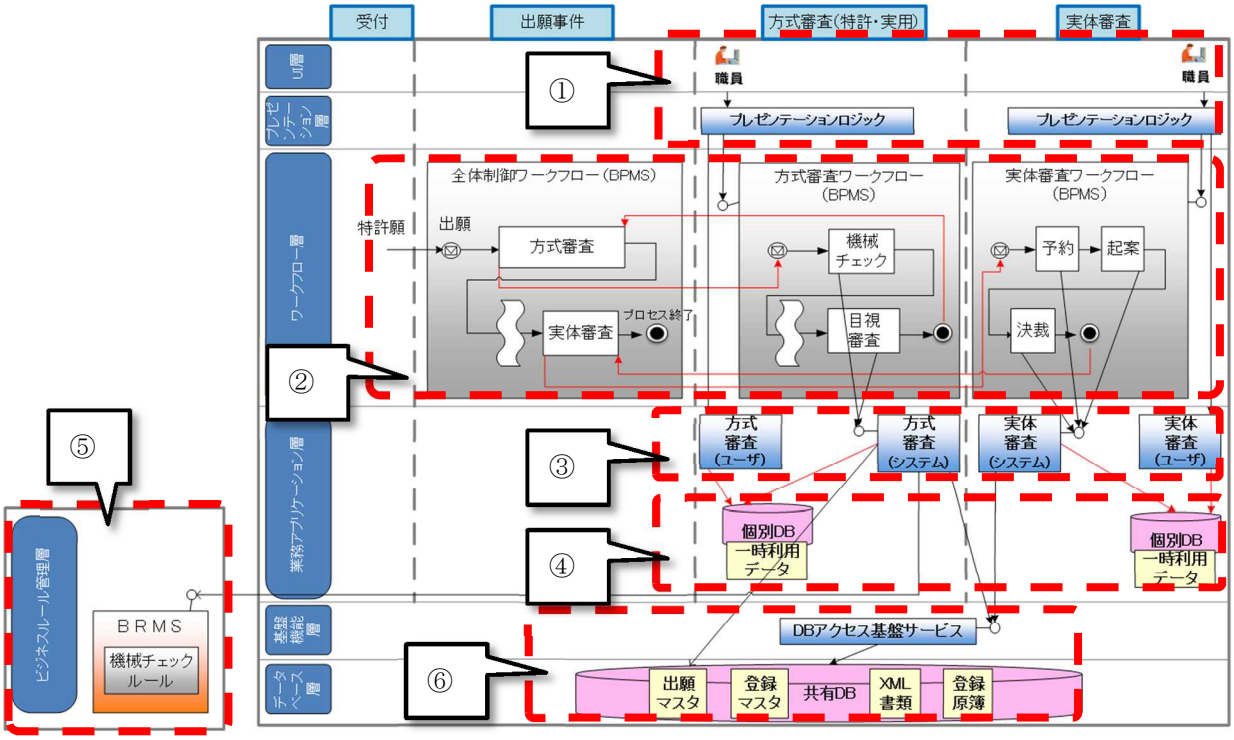
<凡例>
「◎」 :アクセス可(制限なし)
「○」 :アクセス可(制限あり)
「△」 :アクセスパスの特例
「＼」(斜線) :アクセス不可

3.1.1.3.1.4 多階層構造の適用イメージ（参考）

(1)　多階層構造を適用した場合の複数サブシステムにおける連携イメージ

参考として，多階層構造を適用した場合の複数サブシステムにおける連携イメージを，方式審査及び実体審査を題材として下図に示す。

詳細については，『全体システム概念設計書』及び各サブシステムの『個別業務システム概念設計書』を参照のこと。



① 庁職員等が画面を操作し業務を行う。また，画面からワークフローの開始や状況の確認等も行う。
② ワークフローを制御する。
③ 業務アプリケーションを実行する。
④ サブシステム固有の業務データを管理する。
⑤ ビジネスルールを管理，実行する。
⑥ システム全体で共有するデータを管理する。

図 3.1-13 方式審査及び実体審査を題材とした連携イメージ（参考）

上図の②のワークフローを階層化して図示したものを「図 3.1-14 方式審査及び実体審査を題材とした連携イメージ（ワークフローを階層化）（参考）」に示す。

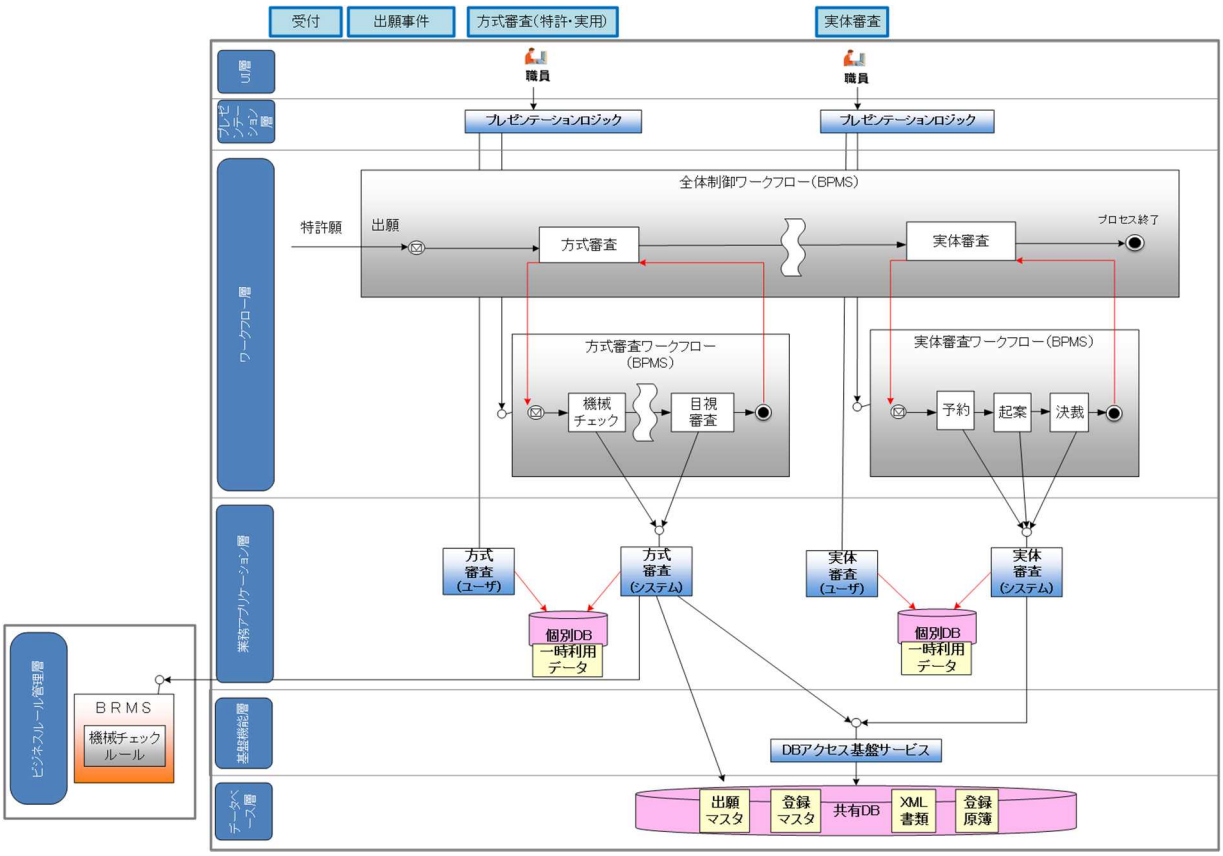


図 3.1-14 方式審査及び実体審査を題材とした連携イメージ(ワークフローを階層化)(参考)

(2)　外部システム連携層を介した内部システムと外部システムとの連携イメージ

参考として，外部システム連携層を介した内部システムと外部システムとの連携イメージを下図に示す。外部システム連携層によりギャップ吸収された結果，外部システムがあたかも刷新化されたシステム（内部システム）のようにエミュレートされる。これにより，呼び出し元から見て，呼び出し先が内部システム，外部システムのいずれであろうとも，内部システムのサービスインタフェースを呼び出すように扱える。したがって，内部システムと外部システム連携層間は，外部システムが刷新され内部サブシステムとなったときのインタフェースで連携が可能となる。

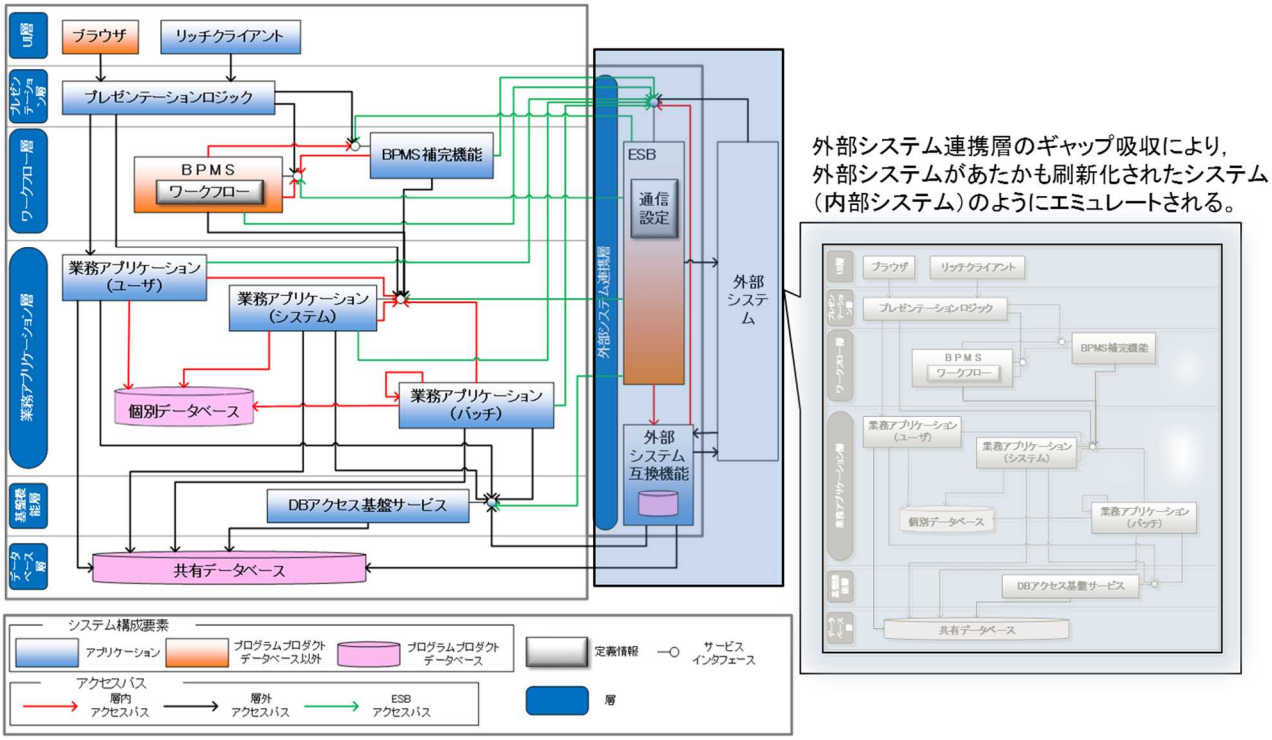


図 3.1-15 内部システムと外部システムとの連携イメージ（参考）

(3)　サブシステムと多階層構造の関連のイメージ

参考として，サブシステムと多階層構造の関連のイメージを下図に示す。下図の平面図として表すものは，サブシステムと当該サブシステムが扱うデータを示す。断面図として表すものは多階層構造を示す。刷新後の特許庁システムは，サブシステム（と当該システムが扱うデータ）及び多階層構造の軸を持つキューブのような関係を持つ。

多階層構造は以下の特徴があると言える。

- UI層から業務アプリケーション層はサブシステム毎に構築する。
- BPMSは各サブシステムを横断的に処理するワークフローを持つ。
- 基盤機能層とデータベース層は，UI層から業務アプリケーション層のサブシステム間で共通的に利用される。
- 外部システム連携層は，「(2)外部システム連携層を介した内部システムと外部システムとの連携イメージ」に示したとおり，外部システムを内部システムとしてエミュレートする。

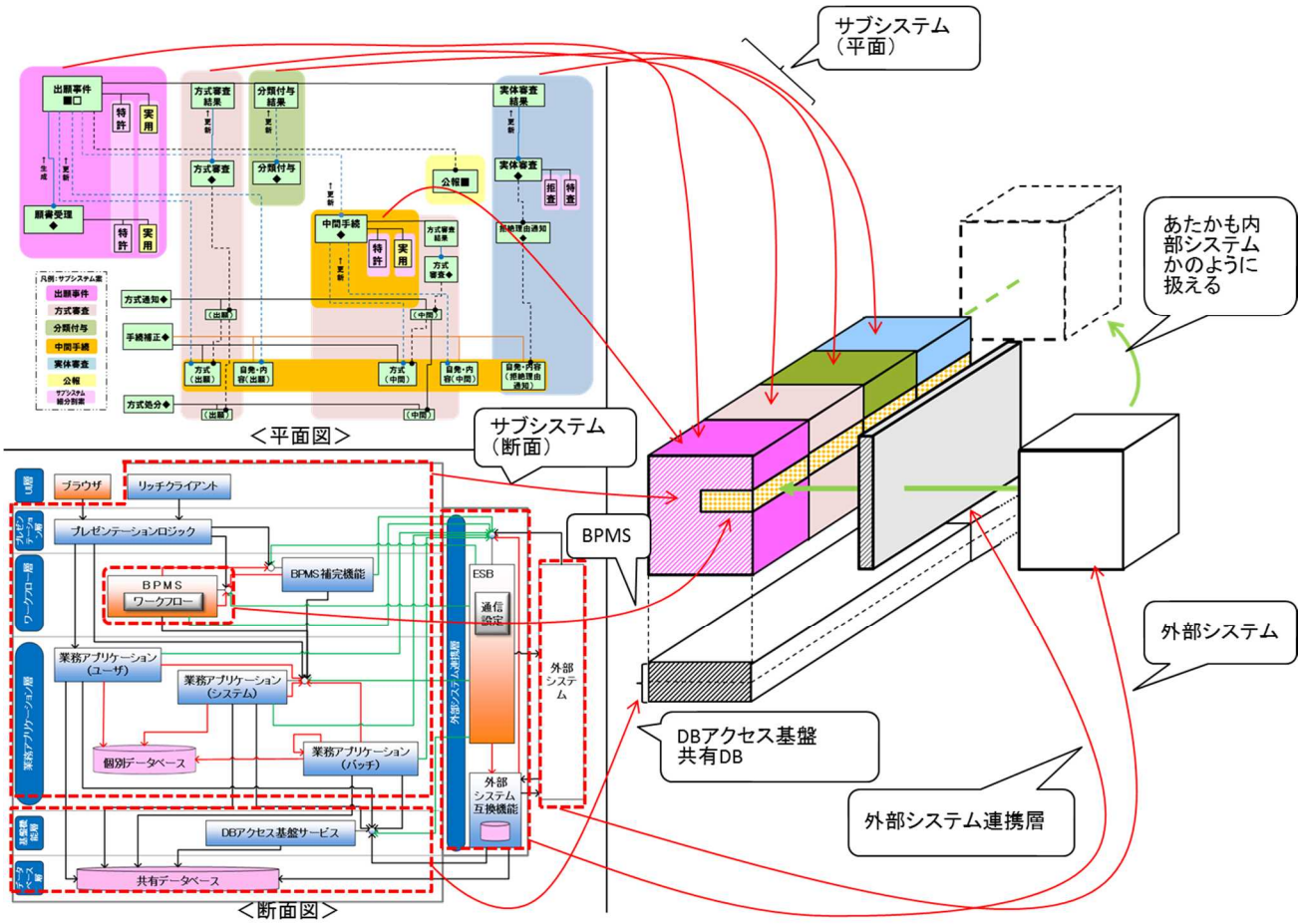


図 3.1-16 サブシステムと多階層構造の関連のイメージ（参考）

3.1.1.3.2 システム構成要素のソフトウェア構成

「3.1.1.3.1 システム構成要素の種類及び責務」の各システム構成要素を論理ノードとして定義し，その論理ノードを構成するソフトウェア構成を定義する。ソフトウェア構成とは，論理ノードにどのソフトウェア及びデータベースの組み合わせが配置されているかを定義したものである。システム構成要素のソフトウェア構成として示す対象を下表に示す。

表 3.1-11 システム構成要素のソフトウェア構成として示す対象

項番	対象	概要説明
1	ソフトウェア	プログラムプロダクト(ミドルウェア製品，プラットフォーム製品等)だけでなく，OSや業務アプリケーションを含んだ広義のソフトウェア。
2	データベース	データベースにおけるデータ部分の要素。 基本的にDBMSとセットで扱われる。
3	論理ノード	ソフトウェア及びデータベースを組み合わせて利用する際の最小構成要素。 ※物理的に分割して配置するとソフトウェアとしての本来の利用目的を達することのできない最小の単位。
4	システム 構成要素	論理ノードから構成される。 なお，以下の組み合わせのようにシステム構成要素間の通信がない場合，物理的に同一の論理ノードとして扱う。 ● プレゼンテーションロジックと業務アプリケーション(ユーザ) ※ただし，静的Webコンテンツ(UI層アプリケーション含む)はWebサーバ上に，動的WebコンテンツはAPサーバ上に配置されることから，それぞれを別の論理ノードとする。 ● ESBと外部システム互換機能

「表 3.1-11 システム構成要素のソフトウェア構成として示す対象」に示したソフトウェア，データベース，論理ノード，システム構成要素の関係を踏まえ，ソフトウェア構成として定義した一覧を以下に示す。

表 3.1-12 ソフトウェア構成一覧

項番	システム構成要素	論理ノード	ソフトウェア又はデータベース
1	ブラウザ	ブラウザ	認証プラグイン／個人認証用ソフトウェア
			Webブラウザ
			OS
2	リッチクライアント	リッチクライアント	UI層アプリケーション(リッチクライアント)
			認証プラグイン／個人認証用ソフトウェア
			OS
3	プレゼンテーションロジック，業務アプリケーション(ユーザ)	Webサーバ	静的Webコンテンツ(UI層アプリケーション含む)
			Webサーバ
			OS
4		業務アプリケーション(ユーザ)サーバ	動的Webコンテンツ
			プレゼンテーションロジック
			業務アプリケーション(ユーザ)
			APサーバ
			OS
5	BPMS	BPMS	BPMS
			APサーバ
			OS
6		BPMS用データベース	DBMS
			BPMS用DB
			OS
7	BPMS補完機能	BPMS補完機能	BPMS補完機能
			APサーバ
			OS

項番	システム構成要素	論理ノード	ソフトウェア又はデータベース
8	業務アプリケーション(システム)	業務アプリケーション(システム)サーバ	業務アプリケーション(システム)
			APサーバ
			OS
9	業務アプリケーション(バッチ)	業務アプリケーション(バッチ)サーバ	業務アプリケーション(バッチ)
			FTPサーバ
			ジョブ管理エージェント
			OS
10	個別データベース	個別データベース	DBMS
			個別DB
			OS
11	BRMS	BRMS	BRMS
			APサーバ
			OS
12	DBアクセス基盤サービス	DBアクセス基盤サービス	DBアクセス基盤サービス
			APサーバ
			OS
13	共有データベース	共有データベース	DBMS
			共有DB
			OS
14	ESB，外部システム互換機能	外部システム連携	外部システム互換機能
			ESB
			OS
15		外部システム連携用データベース	DBMS
			外部システム連携用DB
			OS

以下に，ソフトウェア(プログラムプロダクトと業務AP等の開発対象のアプリケーション)とデータベースの一覧を示す。

表 3.1-13 ソフトウェア(プログラムプロダクト)一覧

項番	プログラムプロダクト	説明
1	BPMS	『別冊3 プログラムプロダクト編』を参照のこと。
2	BRMS	
3	ESB	
4	Webブラウザ	
5	Webサーバ	
6	APサーバ	
7	FTPサーバ	
8	ジョブ管理エージェント	
9	DBMS	
10	認証プラグイン／個人認証用ソフトウェア	

表 3.1-14 ソフトウェア（開発対象のアプリケーション）一覧

項番	開発対象のアプリケーション	補足説明
1	UI層アプリケーション（リッチクライアント）	開発対象のアプリケーションの内容は，「3.1.1.3.1 システム構成要素の種類及び責務」の各システム構成要素の責務を参照のこと。
2	プレゼンテーションロジック	
3	業務アプリケーション（ユーザ）	
4	BPMS補完機能	
5	業務アプリケーション（システム）	
6	業務アプリケーション（バッチ）	
7	DBアクセス基盤サービス	
8	外部システム互換機能	
9	静的Webコンテンツ （UI層アプリケーション含む）[18]	静的Webコンテンツとは，クライアントからのリクエストに応じてそのままレスポンスの一部として返されるもののこと。以下の要素からなる。 ● Webブラウザ上の静的な画面要素となるファイル（HTMLファイル，等） ● Webブラウザ上で動作するアプリケーションの実行ファイル(JavaScriptファイル，等)
10	動的Webコンテンツ[18]	動的Webコンテンツとは，クライアントからのリクエストに応じて動作するアプリケーションが，レスポンスの一部として動的に情報を生成する際に必要とするもののこと。以下の要素からなる。 ● Webブラウザ上の動的な画面要素や情報を生成するためのファイル（JSPファイル，等）

表 3.1-15 データベース一覧

項番	データベース	説明
1	個別DB	「3.1.1.3.1 システム構成要素の種類及び責務」の各システム構成要素の責務を参照のこと。
2	共有DB	
3	BPMS用DB	BPMSが利用するためのデータベース。
4	外部システム連携用DB	外部システム連携で利用するためのデータベース。

ソフトウェア構成は，アーキテクチャの違いが明らかに異なるサブシステム毎の定型タイプを定める。ソフトウェア構成の単位を下表に示す。

表 3.1-16 ソフトウェア構成として示す単位

項番	ソフトウェア構成の単位	ソフトウェア構成の概要	補足説明
1	階層定型化サブシステム （業務遂行・共通リソース系）	サブシステム毎の業務遂行及び共通リソース管理を行うサブシステム。	―
2	階層定型化サブシステム （共有データベース管理）	事件データ・書類データの管理を行うサブシステム。	―
3	インタフェース定型化サブシステム（外部システム連携）	ESB及び外部連携互換機能を備えるサブシステム。	インタフェース定型化サブシステムであるが，ToBeのアーキテクチャ上，新規に追加されたシステム構成要素の部分であるため，一例としてソフトウェア構成図に示す。
4	インタフェース定型化サブシステム（ビジネスルール管理）	BRMSを備えるサブシステム。	

[18] Webコンテンツの配置場所は，実行時に動作する場所（Webブラウザがあるクライアント PC側）ではなく，システム構築を行う際に配置される場所（Web/APサーバ側）とする。

ソフトウェア構成として示す対象及び単位を踏まえ，ソフトウェア構成図を下図に示す。なお，「ブラウザ」はサブシステム内のシステム構成要素ではないため，ソフトウェア構成図上ではシステム構成要素単体で記載している。

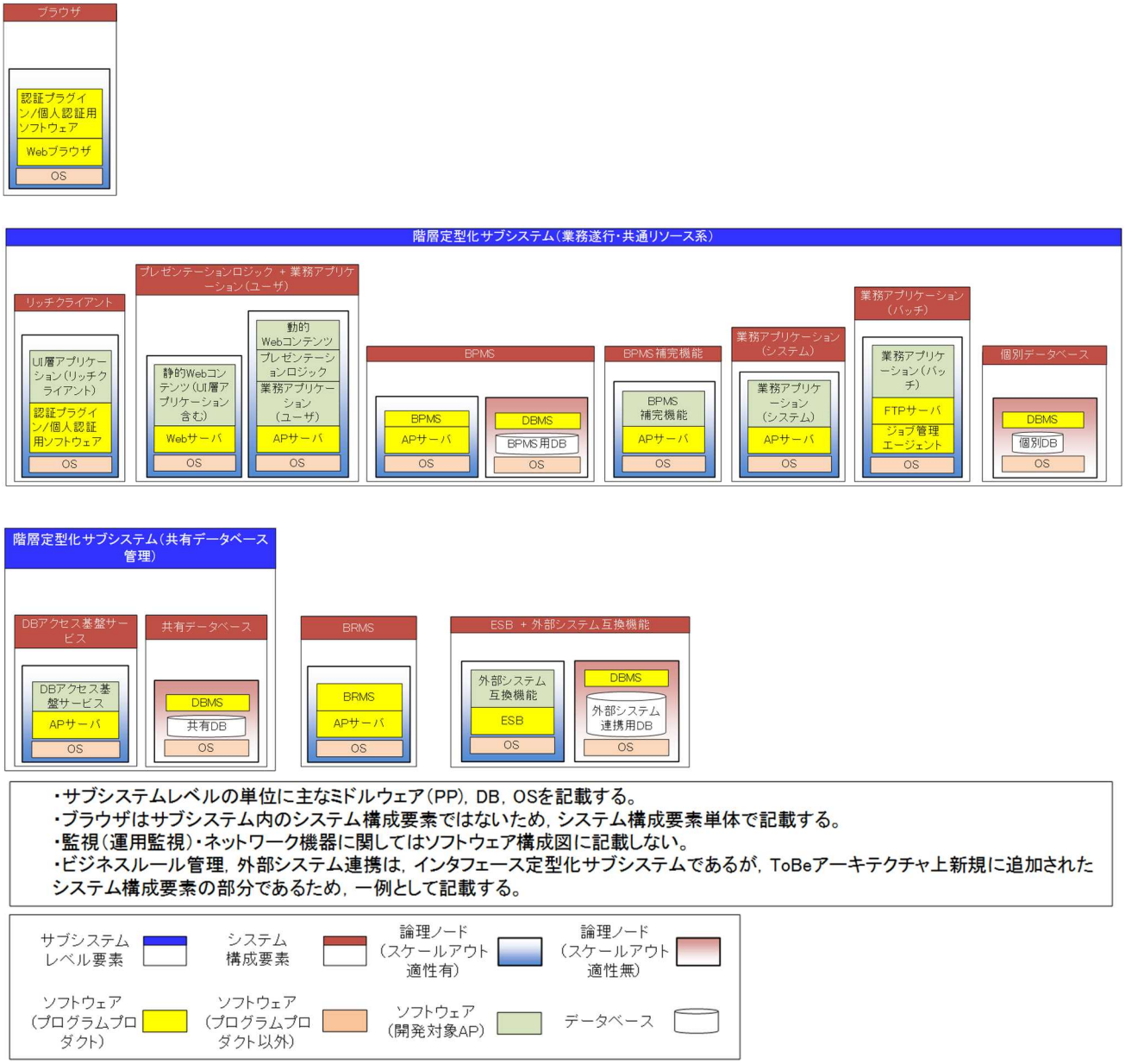


図 3.1-17 ソフトウェア構成図[19]

[19] 論理ノードについては，物理的な配置の際の目安とするために，スケールアウトの適性についても示している。

3.1.2 システムの動的な振る舞い

本章に示す技術方式のルール(設計指針, 規約, 推奨及び特例)は, 以下の章に示す。

- 3.1.2.1 APレイヤの定義及びコンポーネントとの関係
- 3.1.2.2 オンライン処理方式
- 3.1.2.3 バッチ処理方式
- 3.1.2.4 ディレードバッチ処理方式
- 3.1.2.5 帳票処理方式
- 3.1.2.6 サブシステム間連携処理方式

3.1.2.1 APレイヤの定義及びコンポーネントとの関係

目的:	アプリケーション内部のコンポーネントとコンポーネント間の依存関係を定義することにより，サブシステムを構成するシステム構成要素（アプリケーション）[20]の内部構造を定型化，疎結合化し，変更時の影響箇所を具体的に把握することを容易にするほか，アプリケーション内部のコンポーネントの改造による影響範囲の局所化，拡散の抑制を図る。
スコープ:	階層定型化サブシステム
規約1:	オンライン処理方式のコンポーネント間のアクセスは，「図 3.1-19」，「図 3.1-20」，「図 3.1-22」，「図 3.1-23」，「図 3.1-24」に示すアクセスパスとすること。
規約2:	オンライン処理方式のコンポーネントは，「表 3.1-19～表 3.1-23」に定義される単位で作成し，同表に定義されている責務を全うするようコンポーネントの機能を実現すること。
規約3:	バッチ処理方式のコンポーネント間のアクセスは，「図 3.1-31」に示すアクセスパスとすること。
規約4:	バッチ処理方式のコンポーネントは，「表 3.1-29」に定義される単位で作成し，同表に定義されている責務を全うするようコンポーネントの機能を実現すること。
規約5:	ディレードバッチ処理方式のコンポーネント間のアクセスは，「図 3.1-33」に示すアクセスパスとすること。
規約6:	ディレードバッチ処理方式のコンポーネントは，「表 3.1-35」に定義される単位で作成し，同表に定義されている責務を全うするようコンポーネントの機能を実現すること。

オンライン処理方式及びバッチ処理方式のAPレイヤ・コンポーネント定義を示す。サブシステム間連携処理方式のAPレイヤ・コンポーネント定義は，オンライン処理方式，バッチ処理方式及びディレードバッチ処理方式の組み合わせとなる。

3.1.2.1.1 APレイヤの定義及びコンポーネントとの関係

各階層は，責務を下表に示す三つのAPレイヤに分割し定義する。表の記述順序はAPレイヤの上下関係を示しており，上が上位レイヤ，下が下位レイヤを意味している。

コンポーネントは必ずいずれかのAPレイヤに属し，下位レイヤに属するコンポーネントから上位レイヤへ属するコンポーネントに対するアクセスは，禁止する（下位が上位に依存しない作りとする）。

表 3.1-17 APレイヤ

項番	APレイヤ名	責務
1	インタフェース層	システム構成要素（アプリケーション）の機能を外部に提供するためのインタフェースを定義するレイヤ。このレイヤで，通信基盤等，アプリケーションの実装技術を隠蔽する。
2	ロジック層	システム構成要素（アプリケーション）に求められる機能を実現するレイヤ。
3	インフラストラクチャ層	システム構成要素（アプリケーション）に求められる機能を実現するために外部にアクセスする必要がある場合，アクセスするための実装技術を隠蔽するレイヤ。 製品の提供する通信ライブラリの独自APIは，インフラストラクチャ層からのみ呼び出すこと。

[20] 多階層構造図におけるシステム構成要素のアプリケーションで示す部分とブラウザを指す。具体的には，リッチクライアント，プレゼンテーションロジック，業務アプリケーション（ユーザ），業務アプリケーション（システム），DBアクセス基盤サービス，BPMS補完機能，業務アプリケーション（バッチ），ブラウザのことを指す。

3.1.2.2 オンライン処理方式

目的:	オンライン処理方式のデータ保護方式，エラー処理方式，業務継続方式を各サブシステムで異なるものにならないよう定型化することにより，システムの信頼性の向上を図ると共に，システム構成要素（アプリケーション）の機能を画一化し，変更時の影響箇所の局所化を図る。
スコープ:	階層定型化サブシステム
指針1:	オンライン処理方式のデータ保護方式は，「3.1.2.2.3 データの保護方式」に従って設計すること。
指針2:	オンライン処理方式のエラー処理方式は，「3.1.2.2.4 エラー処理方式」に従って設計すること。
指針3:	オンライン処理方式の業務継続方式は，「3.1.2.2.5 業務継続方式」に従って設計すること。

オンライン処理方式は，クライアント等からの処理要求に対するサーバ処理が同期的に全て完遂する処理を基本とする。

ただし，Ajaxを利用し非同期にデータ授受をすることを許容する。

以下，オンライン処理方式について処理方式パターンを示す。

3.1.2.2.1 処理方式パターン

オンライン処理方式における処理方式のパターンを下表に示す。

表 3.1-18 オンライン処理方式の処理方式パターン

項番	分類	説明
1	画面系	ユーザ操作により，画面からネットワークを介してサーバに処理要求を行い，応答を返却する方式。
2	サービス系	BPMS等からの処理要求に対し，サービスとして応答を返却する方式。

3.1.2.2.2 APレイヤ・コンポーネント定義

(1) 画面系

画面系のオンライン処理では，UI層（ブラウザ又はリッチクライアント），プレゼンテーション層（プレゼンテーションロジック），業務アプリケーション層（業務アプリケーション（ユーザ））の三つの階層を構築することでシステム利用者（庁職員等）に機能を提供する。

なお，プレゼンテーション層と業務アプリケーション層の業務アプリケーション（ユーザ）との連携は，レスポンス向上のためメソッド呼び出しとするため，コンポーネント構成は原則，一つにまとめる。

上記を踏まえ，画面系のオンライン処理方式における概念図及びAPレイヤ・コンポーネント構成を下図，下表に示す。なお，概念図中のアクセスパスは代表的なもののみ記載している。システム構成要素間のアクセスパスに関する説明は，「3.1.1.3.1.3 システム構成要素間のアクセスパス」を参照すること。

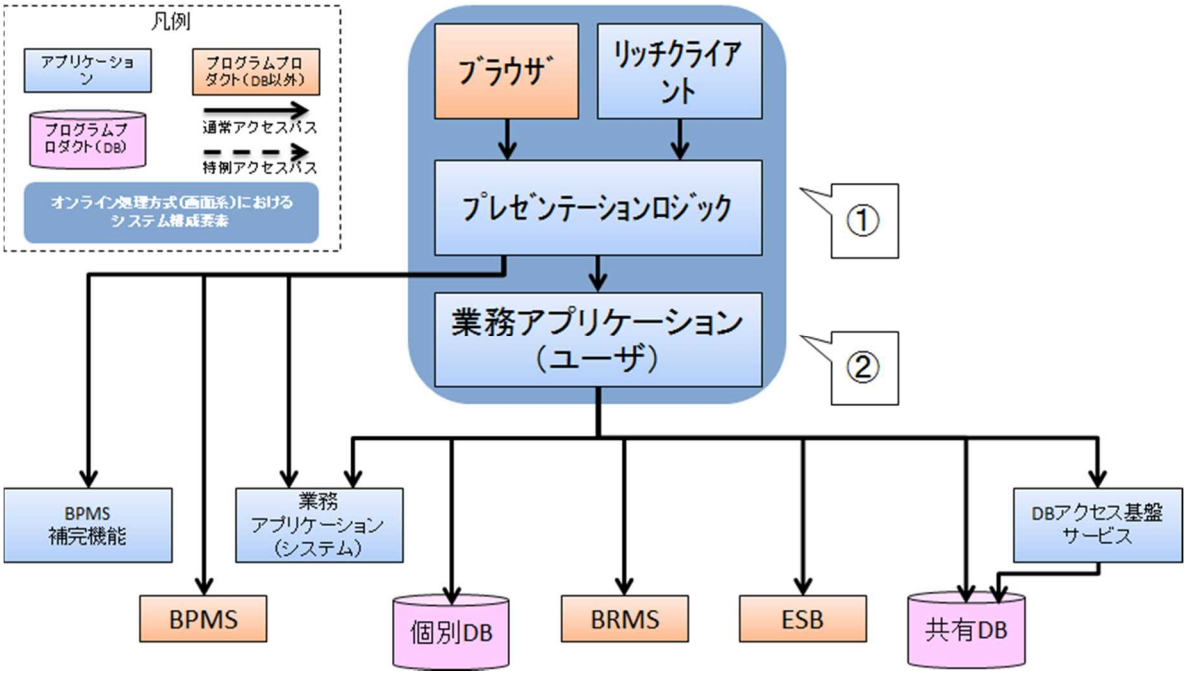


図 3.1-18 オンライン処理方式（画面系）の概念図

① 画面から入力された情報を元に，認証や入力値のチェックを行い，リクエストに応じて下位層のシステム構成要素である業務アプリケーション（ユーザ）やワークフローの呼び出しを行う。また，処理結果の応答を画面へ返却する。

② 画面からの処理要求の内容に応じ，ビジネスロジックの実行のために他のシステム構成要素へアクセスする。

- ブラウザ又はリッチクライアントの場合

図 3.1-19 APレイヤ・コンポーネント定義（画面系・ブラウザ又はリッチクライアント）[21]

[21] 各サブシステムのプレゼンテーションロジックから出力されブラウザ上に配信されたアプリケーション（HTMLやJavaScript）の構成を表している。この図におけるブラウザから見た自サブシステムは，配信元のプレゼンテーションロジックが属するサブシステムのことを指す。

- プレゼンテーションロジック及び業務アプリケーション(ユーザ)の場合

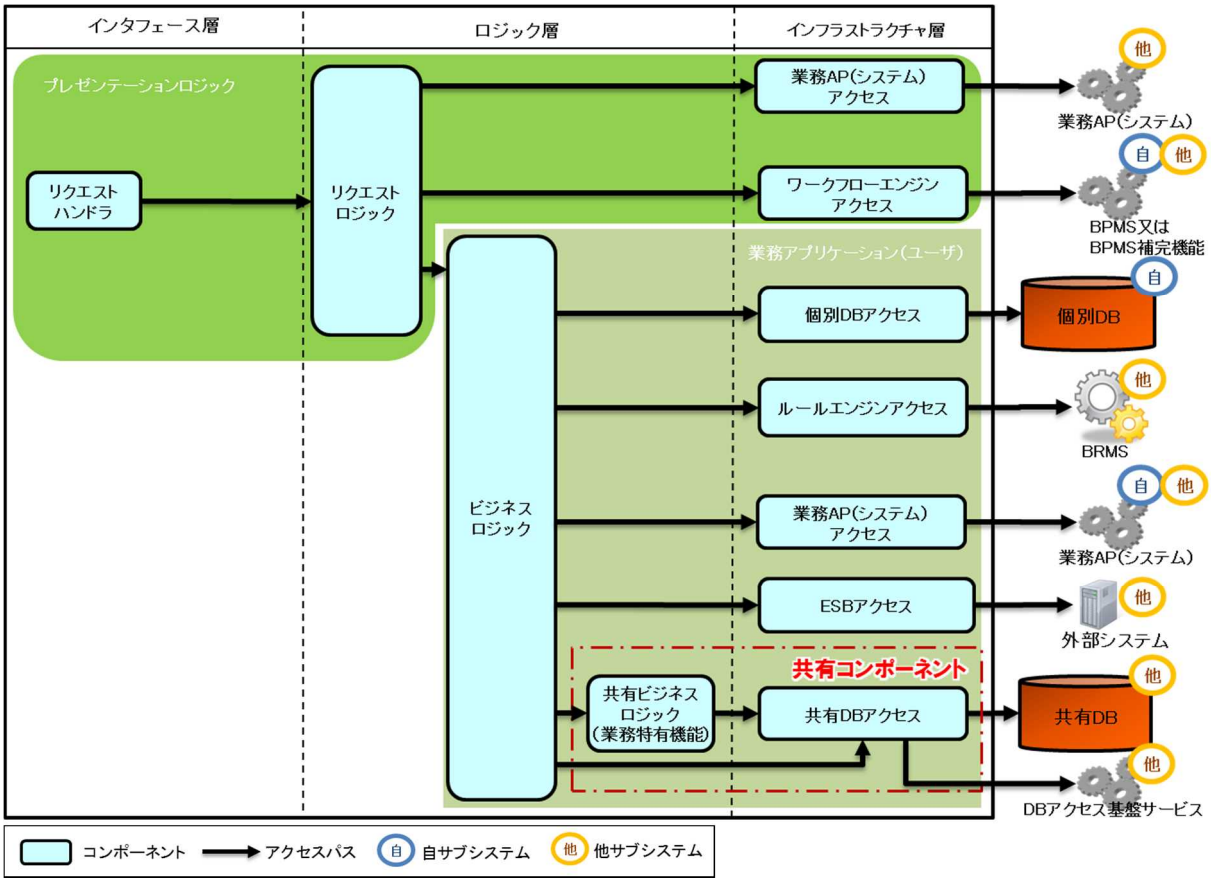


図 3.1-20 APレイヤ・コンポーネント定義(画面系・プレゼンテーションロジック及び業務アプリケーション(ユーザ))

表 3.1-19 ブラウザ又はリッチクライアントの責務

項番	コンポーネント	配置するAPレイヤ	責務
1	画面	インタフェース層	システム利用者が操作するUIを提供する。また,入力規制(桁数制限等)等,入力補助も行う。
2	イベントハンドラ	インタフェース層	画面を操作した際に発生するアクションイベントを制御し,画面ロジックの呼び出し等を行う。
3	画面ロジック	ロジック層	クライアント側で可能な入力チェック(必須チェック,フォーマットチェック等)や画面内の制御,プレゼンテーションロジックからの表示制御等を行う。また,サーバでの処理が必要な場合は,サーバへ送信する要求データの組み立ても行う。
4	プレゼンテーションロジックアクセス	インフラストラクチャ層	プレゼンテーションロジックの呼び出しを行う。また,呼び出す際の固有の通信技術や手順を上位層から隠蔽する。

表 3.1-20 プレゼンテーションロジック及び業務AP(ユーザ)の責務

項番	コンポーネント	配置するAPレイヤ	責務
1	リクエストハンドラ	インタフェース層	プレゼンテーションロジック呼び出しのインタフェースの仕様を定め,認証チェックやインタフェースの入力チェック等(必須チェック,フォーマットチェック等),リクエストロジックやビジネスロジックを呼び出す上での前提条件を満たすことを保証する。 なお,入力チェックは,クライアントでも実施しているものであってもセキュリティ上,サーバでも再度実施する。 また,クライアント構成に応じた情報の返却(HTML,XML電文等)を行う。

項番	コンポーネント	配置するAPレイヤ	責務
2	リクエストロジック	ロジック層	リクエストに応じて下位層のシステム構成要素である業務アプリケーション（ユーザ）や業務アプリケーション（システム），ワークフローの呼び出しを行う。 また，画面に受け渡す値の生成を行う。
3	ビジネスロジック	ロジック層	業務アプリケーション（ユーザ）として提供する主たる機能を実装するコンポーネント。 必要に応じてインフラストラクチャ層のコンポーネントを呼び出し，参照・判定・更新といった処理の組み立てを行う。 また，業務チェック（DB突合チェック等）も行う。
4	共有ビジネスロジック（業務特有機能）	ロジック層	ビジネスロジックのうち，複数の業務アプリケーションで共有し，かつ特許庁業務特有の機能（期間計算等）を提供するコンポーネント。
5	業務AP（システム）アクセス	インフラストラクチャ層	業務アプリケーション（システム）が提供するサービスの呼び出しを行う。アクセスパスの制限事項については「3.1.1.3.1.3 (2)」参照。業務アプリケーション（システム）が提供するサービスを呼び出す際の通信技術や手順を上位層から隠蔽する。
6	ワークフローエンジンアクセス	インフラストラクチャ層	BPMS又はBPMS補完機能の呼び出しを行う。 ロジック層に対して汎用的なAPIを提供するために製品固有のAPIの呼び出しを上位層から隠蔽する。
7	共有DBアクセス	インフラストラクチャ層	共有データベースへの参照・更新機能を提供するコンポーネント。本コンポーネントは，特実記録原本管理システムの設計開発基準に記載されている「基盤API」を指す。詳細は『DBアクセス基盤利用者向けガイドライン － 6．基盤API』を参照のこと。
8	個別DBアクセス	インフラストラクチャ層	個別データベースへの参照・更新機能を提供するコンポーネント。ロジック層に対して汎用的なAPIを提供するために製品固有のデータアクセス技術，永続化手段を上位層から隠蔽する。 なお，横断的に利用する共通的な機能をコンポーネントとして提供し，データベースアクセス処理毎に開発が必要なSQL定義やエンティティクラス等はコンポーネントとは別に管理する。
9	ルールエンジンアクセス	インフラストラクチャ層	BRMSへルールの実行を要求する。 ビジネスルール管理サブシステムのサービス呼び出しを行う。 ロジック層に対してBRMSの呼び出し方式を隠蔽する。
10	ESBアクセス	インフラストラクチャ層	外部システムの呼び出しを行う。外部システムを呼び出す際の通信技術や手順を上位層から隠蔽する。

(2)　サービス系

サービス系のオンライン処理は，業務アプリケーション（システム），DBアクセス基盤サービス及びBPMS補完機能が機能を提供する。

上記を踏まえ，サービス系のオンライン処理方式における概念図及びAPレイヤ・コンポーネント構成を下図，下表に示す。なお，概念図中のアクセスパスは代表的なもののみ記載している。システム構成要素間のアクセスパスに関する説明は，「3.1.1.3.1.3 システム構成要素間のアクセスパス」を参照すること。

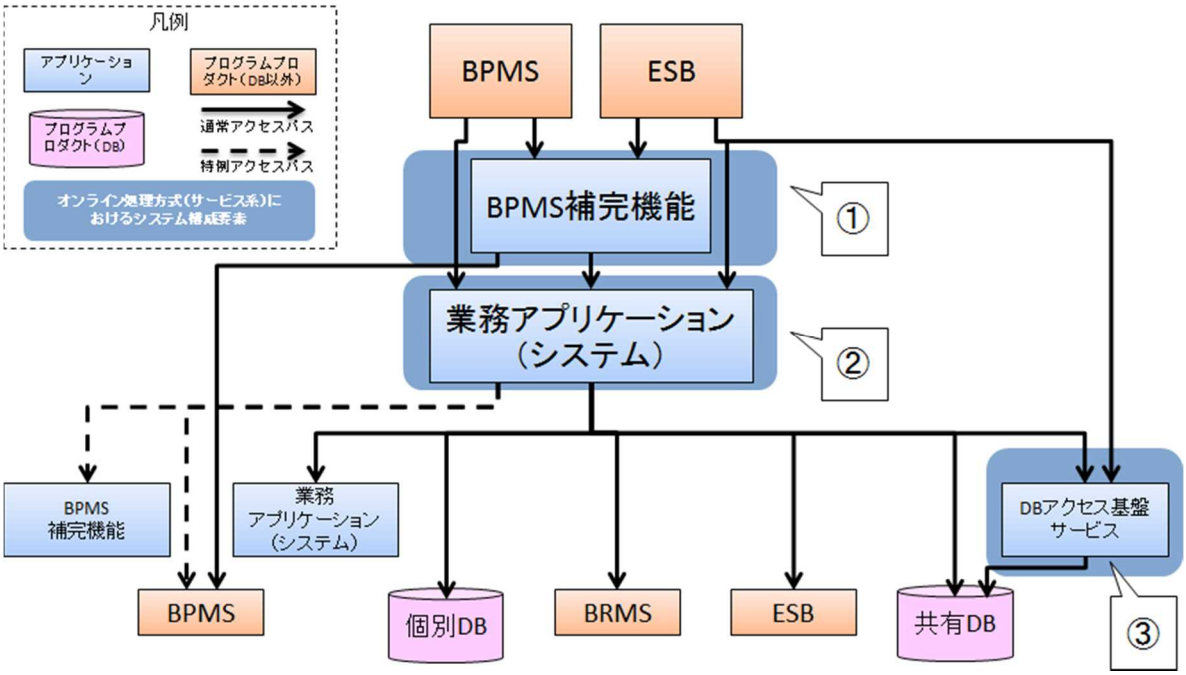


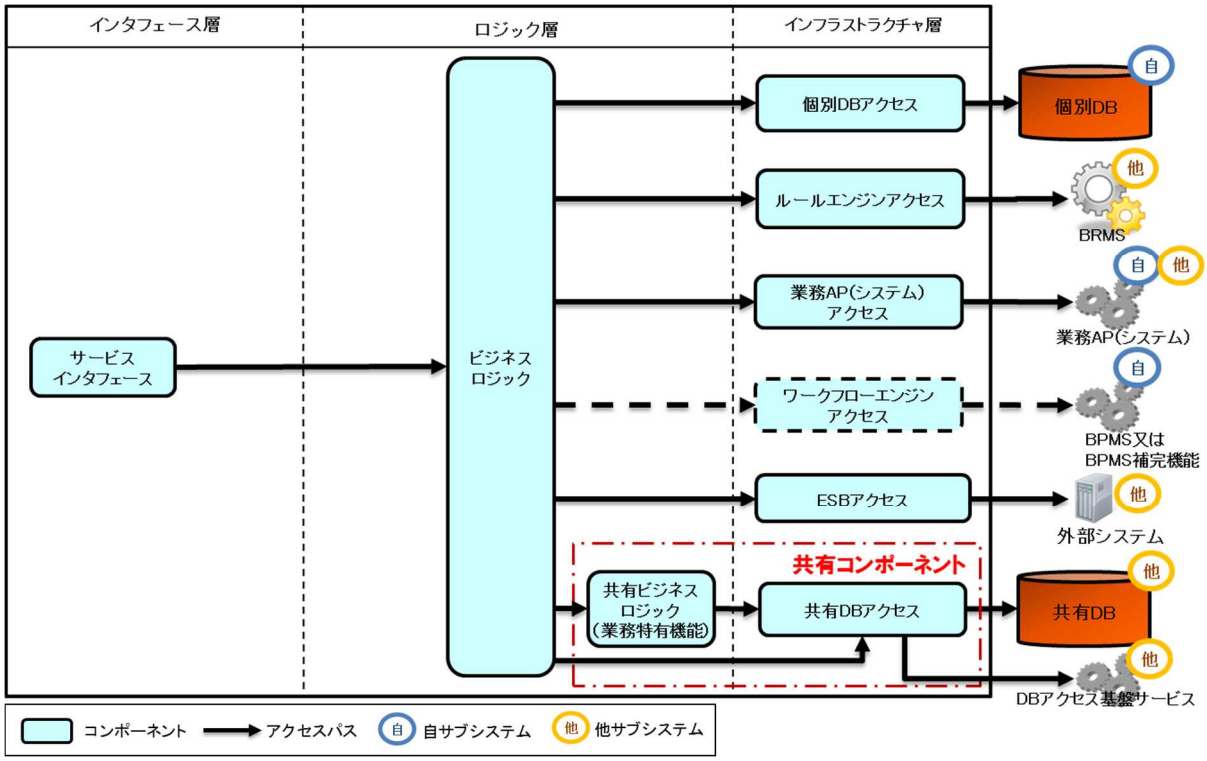
図 3.1-21 オンライン処理方式（サービス系）の概念図

① BPMS補完機能は，ワークフローの制御に必要な情報の参照，更新を行うため，他のシステム構成要素へアクセスする。また，サービス要求元からの処理要求の内容に応じ，BPMSの呼び出しを行う。

② サービス要求元からの処理要求の内容に応じ，ビジネスロジックの実行のために他のシステム構成要素へアクセスする。

③ DBアクセス基盤サービスは，共有データベースへアクセスするためのインタフェースを提供し，共有データベースに格納されているデータに対する参照，更新処理を行う。

- 業務アプリケーション(システム)の場合



※「ワークフローエンジンアクセス」は，アクセスパスの特例において利用を許容する。
点線の矢印は，特例のアクセスパスを示している。
詳細は，「3.1.1.3.1.3 (4)アクセスパスの特例」を参照のこと。

図 3.1-22 APレイヤ・コンポーネント定義(サービス系・業務アプリケーション(システム))

- DBアクセス基盤サービスの場合

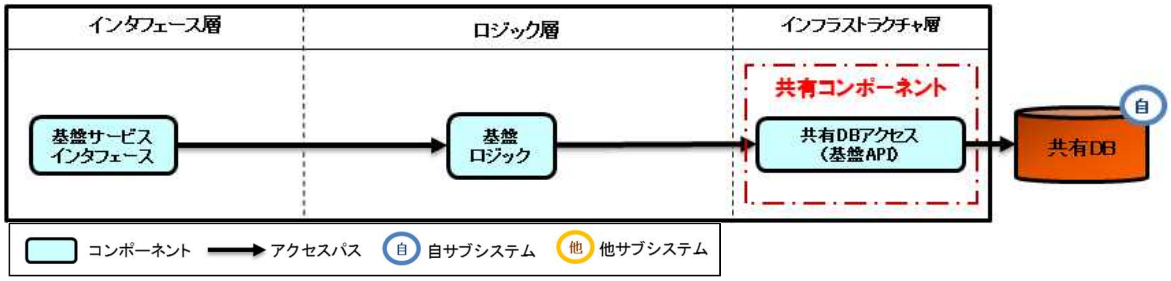


図 3.1-23 APレイヤ・コンポーネント定義(サービス系・DBアクセス基盤サービス)

- BPMS補完機能の場合

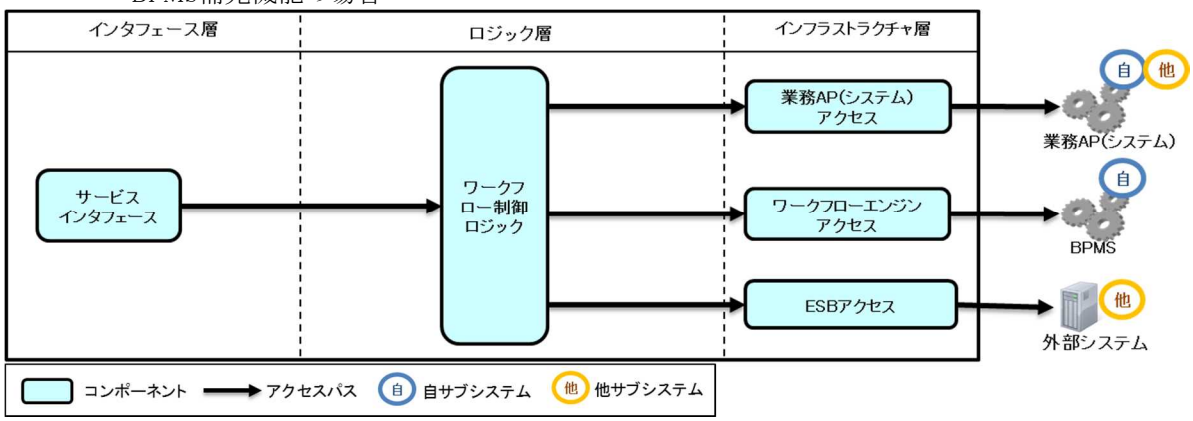


図 3.1-24 APレイヤ・コンポーネント定義(サービス系・BPMS補完機能)

表 3.1-21 業務AP(システム)の責務

項番	コンポーネント	配置するAPレイヤ	責務
1	サービス インタフェース	インタフェース層	サービス起動のインタフェースの仕様を定め，インタフェースの入力チェック等，ビジネスロジックを呼び出す上での前提条件を満たすことを保証する。 また，取得したデータの形式を変換して返却する。
2	ビジネスロジック	ロジック層	業務アプリケーション(システム)として提供する主たる機能を実装するコンポーネント。 必要に応じてインフラストラクチャ層のコンポーネントを呼び出し，参照・判定・更新といった処理の組み立てを行う。 また，業務チェック(DB突合チェック等)も行う。
3	共有ビジネスロジック (業務特有機能)	ロジック層	ビジネスロジックのうち，複数の業務アプリケーションで共有し，かつ特許庁業務特有の機能(期間計算等)を提供するコンポーネント。
4	共有DBアクセス	インフラストラクチャ層	共有データベースへの参照・更新機能を提供するコンポーネント。本コンポーネントは，特実記録原本管理システムの設計開発基準に記載されている「基盤API」を指す。詳細は『DBアクセス基盤利用者向けガイドライン － 6．基盤API』を参照のこと。
5	個別DBアクセス	インフラストラクチャ層	個別データベースへの参照・更新機能を提供するコンポーネント。ロジック層に対して汎用的なAPIを提供するために製品固有のデータアクセス技術，永続化手段を上位層から隠蔽する。 なお，横断的に利用する共通的な機能をコンポーネントとして提供し，データベースアクセス処理毎に開発が必要なSQL定義やエンティティクラス等はコンポーネントとは別に管理する。
6	ルールエンジン アクセス	インフラストラクチャ層	BRMSへルールの実行を要求する。 ビジネスルール管理サブシステムのサービス呼び出しを行う。 ロジック層に対してBRMSの呼び出し方式を隠蔽する。
7	業務AP(システム) アクセス	インフラストラクチャ層	業務アプリケーション(システム)が提供するサービスの呼び出しを行う。アクセスパスの制限事項については「3.1.1.3.1.3 (2)」参照。業務アプリケーション(システム)が提供するサービスを呼び出す際の通信技術や手順を上位層から隠蔽する。
8	ワークフローエンジン アクセス	インフラストラクチャ層	BPMS又はBPMS補完機能の呼び出しを行う。 ロジック層に対して汎用的なAPIを提供するために製品固有のAPIの呼び出しを上位層から隠蔽する。 なお，特異な条件時のアクセスとして許容される。詳細は，「3.1.1.3.1.3 (4)アクセスパスの特例」を参照のこと。
9	ESBアクセス	インフラストラクチャ層	外部システムの呼び出しを行う。外部システムを呼び出す際の通信技術や手順を上位層から隠蔽する。

表 3.1-22 DBアクセス基盤サービスの責務

項番	コンポーネント	配置するAPレイヤ	責務
1	基盤サービス インタフェース	インタフェース層	サービス起動のインタフェースの仕様を定め，インタフェースの入力チェック等，基盤ロジックを呼び出す上での前提条件を満たすことを保証する。 また，取得したデータの形式を変換して返却する。
2	基盤ロジック	ロジック層	共有データベースからデータを取得し，返却用のデータを組み立てる。
3	共有DBアクセス （基盤API）	インフラストラクチャ層	共有データベースへの参照・更新機能を提供するコンポーネント。本コンポーネントは，特実記録原本管理システムの設計開発基準に記載されている「基盤API」を指す。詳細は『DBアクセス基盤利用者向けガイドライン － 6．基盤API』を参照のこと。

表 3.1-23 BPMS補完機能の責務

項番	コンポーネント	配置するAPレイヤ	責務
1	サービス インタフェース	インタフェース層	サービス起動のインタフェースの仕様を定め，インタフェースの入力チェック等，ビジネスロジックを呼び出す上での前提条件を満たすことを保証する。 また，取得したデータの形式を変換して返却する。
2	ワークフロー制御 ロジック	ロジック層	BPMS補完機能として提供する主たる機能を実装するコンポーネント。 必要に応じてインフラストラクチャ層のコンポーネントを呼び出し，参照・判定・更新といった処理の組み立てを行う。
3	ワークフローエンジン アクセス	インフラストラクチャ層	BPMSの呼び出しを行う。 ロジック層に対して汎用的なAPIを提供するために製品固有のAPIの呼び出しを上位層から隠蔽する。
4	業務AP（システム） アクセス	インフラストラクチャ層	業務アプリケーション（システム）が提供するサービスの呼び出しを行う。アクセスパスの制限事項については「3.1.1.3.1.3 (2)」参照。業務アプリケーション（システム）が提供するサービスを呼び出す際の通信技術や手順を上位層から隠蔽する。
5	ESBアクセス	インフラストラクチャ層	外部システムの呼び出しを行う。外部システムを呼び出す際の通信技術や手順を上位層から隠蔽する。

3.1.2.2.3 データの保護方式

(1) アプリケーションによるデータの保護方式

ビジネスプロセスのアクティビティは業務として意味のある最小の単位となるため，データ整合性をアクティビティの単位で確保する必要がある。

一回のリクエストにおける単一データベース内のACID特性を確保する場合は，RDBMSの「システムトランザクション」機能を使用する（ACID特性が求められる単位にトランザクションを制御し，エラー発生時はロールバックする）。

リクエストをまたがって同一のデータに対して同時に複数のアクセスがある場合や複数データベースをまたがるアクセスがある場合，データ整合性を確保するためにアプリケーションにおいて下記の排他制御方式を使用する。なお，複数データベースをまたがるアクセスがある場合，2フェーズコミット方式によりデータ整合性を確保する方法もあるが，ベンダ・サポートを受けるにはトランザクションに参加する製品が全て同一ベンダである必要がある等の制約があるため，使用しない。

表 3.1-24 排他制御方式

項番	排他制御方式	処理概要	一般的な利用ケース
1	悲観的排他制御	あるデータ更新作業の開始～終了時点まで，他からの更新を防ぐ排他制御	先に処理を開始した方の更新を優先したい場合
2	楽観的排他制御	データ更新時に競合をチェックし，更新してよいかを判断する排他制御	処理の開始が先か後かに関わらず，更新が早い方を優先したい場合

BPMSを介したデータ授受を行う場合は以下を考慮すること。考慮点の具体例については，後述の「(2) アプリケーションによるデータ整合性の確保例」を参照すること。

- 他のビジネスプロセスインスタンスによってデータベースの値が更新され，それを自ビジネスプロセスインスタンスが知る必要がある場合，BPMSが通知することで，データ不整合が起きないよう制御すること。
- 僅かなタイムラグでデータ不整合が起きる可能性があるため，データベース更新時は楽観排他又は悲観排他を行うこと。

また，APレイヤ・コンポーネント定義における「ワークフローエンジンアクセス」（BPMSのステータス管理機能によるアクティビティのステータスの更新），「共有DBアクセス（基盤API）」（共有データベースのデータの更新），「個別DBアクセス」（個別データベースのデータの更新）の整合性を確保するための対策を，アプリケーションで取ること。前述のように，2フェーズコミットを使用しない方針により，これらの整合性についてもアプリケーションで取る必要がある。BPMSのアクティビティからサービスを呼び出す処理方式パターンにおける例を以下に示す。

- サービス内での「共有DBアクセス（基盤API）」と「個別DBアクセス」の整合性を取るために，更新の順序を考慮し，個々のデータベースに対してコミットを行うようにする（以下の「図 3.1-25」に処理イメージを示す）。エラーが発生したときは，後続のアクティビティを進ませないようにエラーを返すようにする（以下の「図 3.1-26」に処理イメージを示す）。

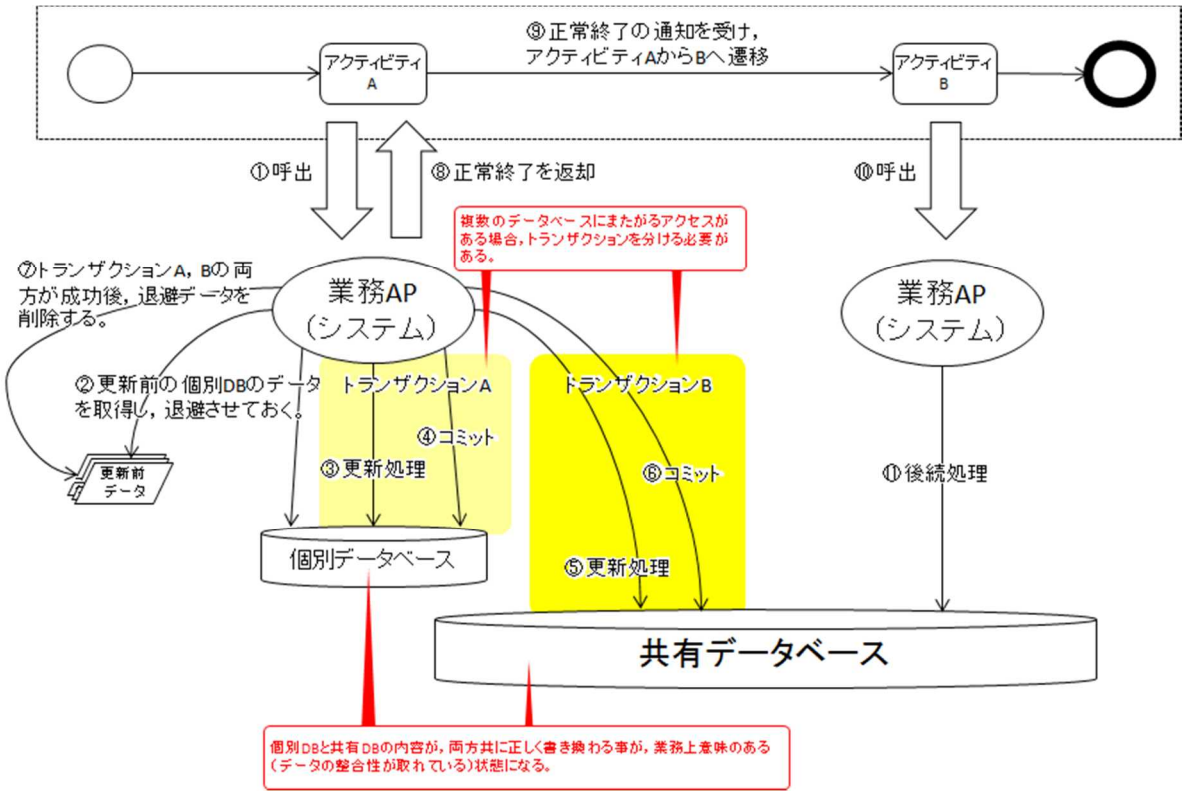


図 3.1-25 アプリケーションによるデータの保護方式

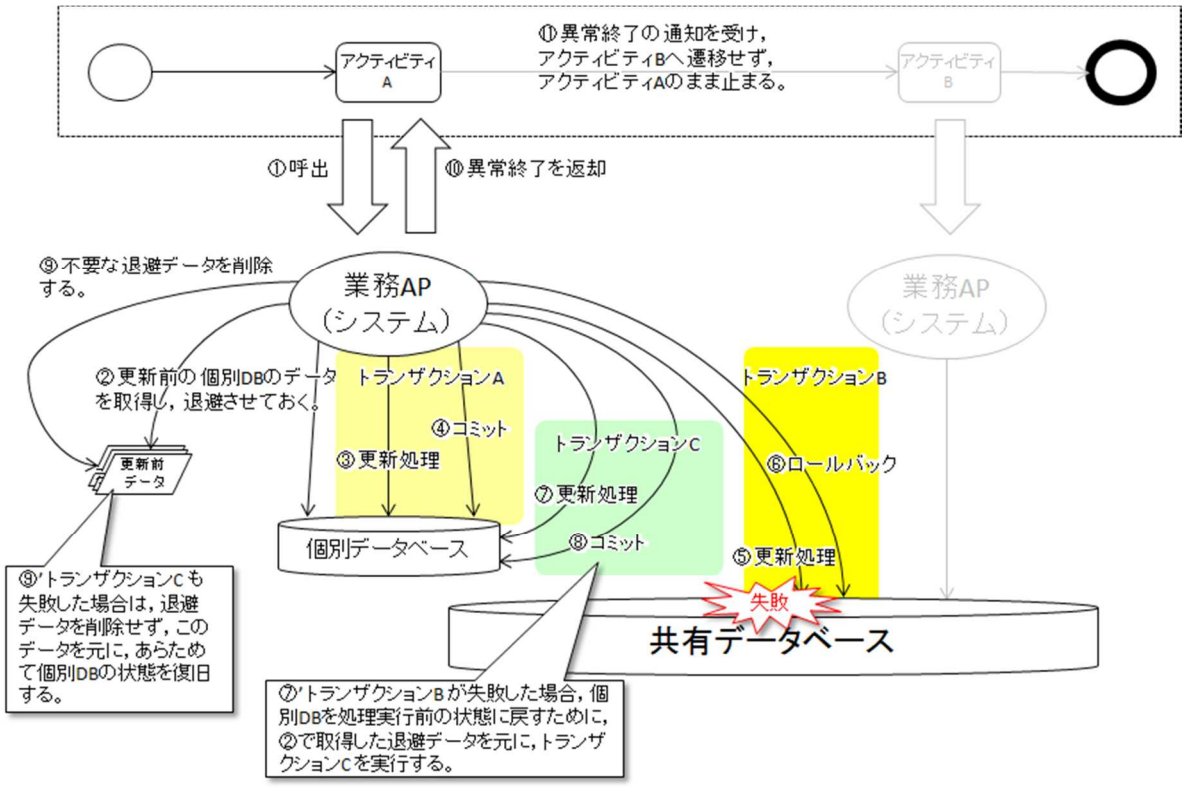


図 3.1-26 データベースアクセスエラー時におけるデータの整合性の確保

- 「ワークフローエンジンアクセス」とサービス内での「共有DBアクセス（基盤API）」又は「個別DBアクセス」の整合性を取るために，BPMSでのアクティビティから同期的にサービスを呼び出し，サービスの結果応答を得るまではプロセストークン（プロセスにおける現在の実行ポイント）を次のノードに進ませないようにする（「図 3.1-27」）。これら一連の処理はBPMSの機能によって行われるが，ここでの不整合発生の可能性はBPMSのシステム障害が由来のものでありサービス内の処理によって回復することは出来ないため，不整合が発生した場合はBPMSのステータスを変更する等の運用対処を行う必要がある。

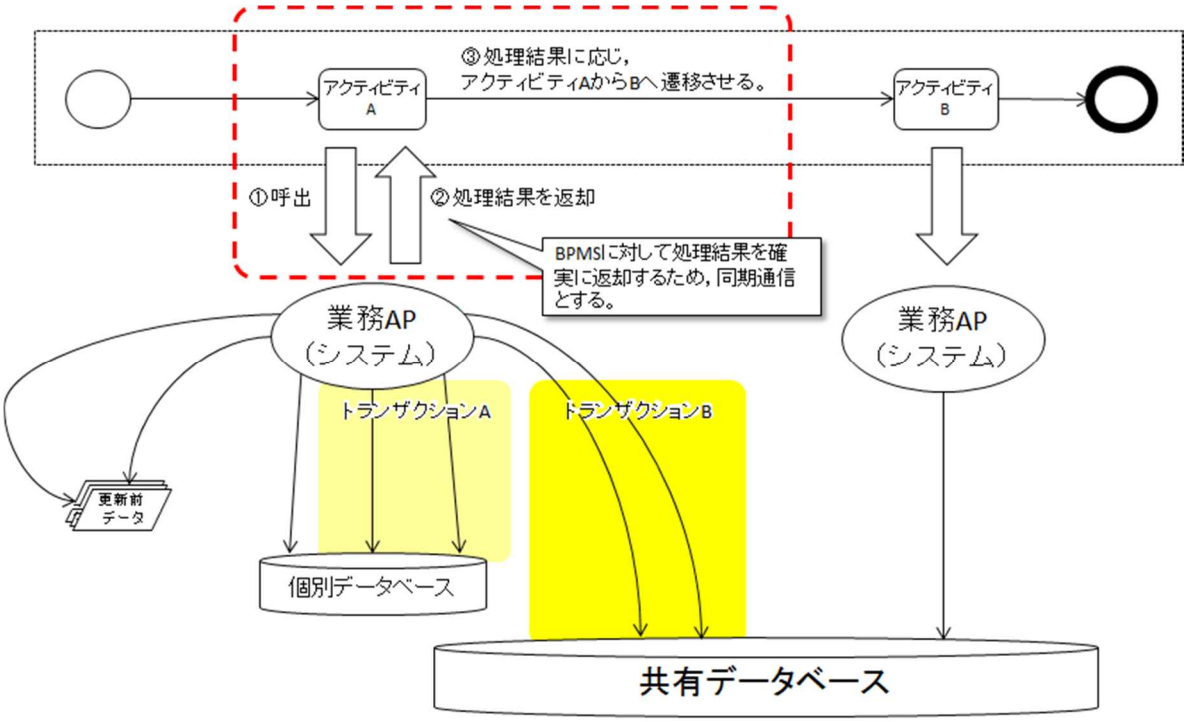


図 3.1-27 BPMSのアクティビティとアプリケーションとの連携

(2) アプリケーションによるデータ整合性の確保例

データベースを介したデータの授受において，実際にデータ不整合が起こり得る例として，「実体審査中の出願取下」を想定する。

下記図のように，実体審査を行っている最中（図中の①）に取下書類が提出された場合，取下書類の方式審査完了時にデータベースが取下げ状態として更新される（図中の②）想定だが，何も対策を講じなければ，実体審査完了時に審査周辺サービスがデータベースを審査完了状態として更新してしまう（図中の③）。

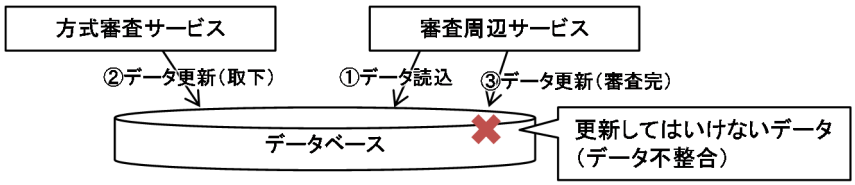


図 3.1-28 実体審査中の出願取下の処理イメージ

ただし，ToBeアーキテクチャにおいては，BPMSが業務割り込みも含めて業務開始契機を適切に制御するため，取下書類が提出された時点で審査周辺サービスにも通知がなされる。そのため，BPMSが存在するアーキテクチャでは上記のようなデータ不整合は発生しないと言える。

ToBeアーキテクチャにおける実体審査中の出願取下の処理イメージを，以下の図に示す。

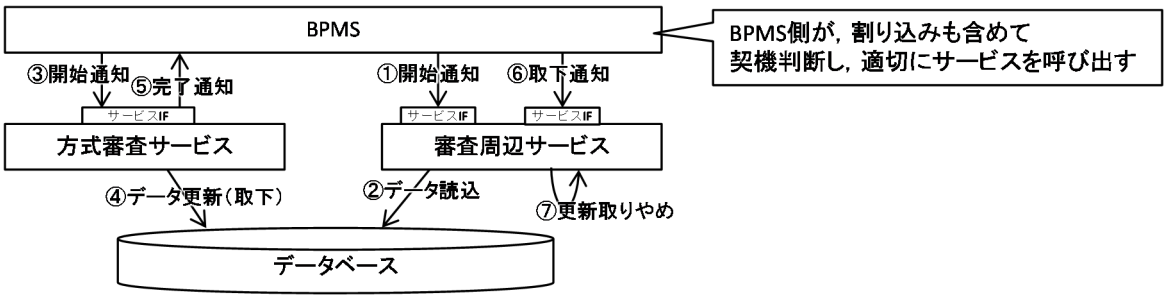


図 3.1-29 ToBeアーキテクチャにおける実体審査中の出願取下の処理イメージ

上記の図のとおり，BPMSを用いた業務開始契機の制御方法を採る事により，基本的には不整合が起きる状況を防ぐことができるが，上記図中の④～⑥の処理時間は僅かながら存在し，その間にデータベースに対する更新（データ不整合）が起きてしまう可能性がある。

そのような可能性を極力排除するためには，前述のBPMSを用いた業務開始契機の制御方法に加え，データベース更新時（例示でいう審査周辺サービスがデータベースを更新する（図中の⑦）際）に，何らかの排他制御を設ける必要がある。

排他制御の方法には，前述のとおり「楽観的排他制御」と「悲観的排他制御」が存在するが，それらは場合に応じて使い分ける必要がある。本ケースにおいては，先に処理が開始した審査周辺サービスではなく，後から処理が開始した方式審査サービスの更新処理を優先する必要があり，このような場合には楽観排他制御を採用する（例えば，データベース内にバージョン番号（履歴番号）を用意し，上記図において②で読み込んだ際のバージョン番号と，データベース更新時のバージョン番号を比較，一致していなければ更新エラーとする））。

このように，BPMSだけでなく排他制御も併用することで，残った僅かなタイミングで不整合が発生する可能性を防ぐことが可能となる。

3.1.2.2.4 エラー処理方式

オンラインで発生するエラーの種類を下表に示す。

表 3.1-25 エラーの種類

項番	エラーの種類	定義	例
1	業務エラー	業務設計(仕様)の中で想定しているエラー	● 入力チェックエラー ● 認証・認可エラー 等
2	システムエラー	業務設計(仕様)の中で想定していないエラー	● OSから通知されるエラー ● ミドルウェアから通知されるエラー 等

上記のエラーが発生した場合，以下の対応をするものとする。

表 3.1-26 エラー発生時の対応

項番	対応	目的
1	エラーの詳細をログ等に出力する。	データの保護や，障害解析性を向上する。
2	データを整合性のある状態に戻す。	データの整合性を確保する。 詳細は，「3.1.2.2.3 データの保護方式」を参照のこと。
3	エラーの発生内容をクライアントに応答する。	システム利用者や運用者にエラーを知らせる。

なお，システムエラーといった，業務設計(仕様)の中で想定されていないエラーを扱う処理は，通常の業務処理のプログラム中に混在して記述すると処理の流れの見通しが悪くなり，保守性が低下する。このような問題を回避するため，エラー処理を集約的に定義・実行できる仕組み(集約エラーハンドラ)を用いることとする。

また，各モジュールの責任範囲外の値が入力された場合，処理を継続せずに例外を発生させることとする。

3.1.2.2.5 業務継続方式

オンライン処理方式では再実行により業務を継続する。詳細を下表に示す。

表 3.1-27 業務継続方式

項番	分類	エラーの種類	業務継続方式
1	画面系	業務エラー	● 画面に表示されるメッセージを表示することでシステム利用者に通知する。 ● メッセージは，エラーの原因や再入力方法が分かるようにする。 ● システム利用者が再実行する。
2		システムエラー	● 業務アプリケーション等がログを出力する。 ● ログは，エラーの対象を特定できるようにする。 ● データ不整合が発生している場合は，運用者がデータを整合性のある状態に復旧する。 ● 復旧後，システム利用者又は運用者が再実行する。
3	サービス系	業務エラー	
4		システムエラー	

3.1.2.3 バッチ処理方式

目的:	バッチ処理方式のデータ保護方式，エラー処理方式，業務継続方式を各サブシステムで異なるものにならないよう定型化することにより，システムの信頼性の向上を図ると共に，システム構成要素（アプリケーション）[22]の機能を画一化し，変更時の影響箇所の局所化を図る。
スコープ:	階層定型化サブシステム
指針1:	バッチ処理方式のデータ保護方式は，「3.1.2.3.3 データの保護方式」に従って設計すること。
指針2:	バッチ処理方式のエラー処理方式は，「3.1.2.3.4 エラー処理方式」に従って設計すること。
指針3:	バッチ処理方式の業務継続方式は，「3.1.2.3.5 業務継続方式」に示される業務継続方式のいずれかを採用した設計とすること。

以下，バッチ処理方式について処理方式パターンを示す。

3.1.2.3.1 処理方式パターン

バッチ処理方式における処理方式のパターンを下表に示す。

表 3.1-28 バッチ処理方式の処理方式パターン

項番	分類	説明
1	複数件一括処理	多量のデータを取りまとめ，一定期間あるいはデータ量毎に一度にまとめて処理する方式。一括処理単位は排他的に処理を行う。
2	単件一括処理	単件の処理を一度にまとめて処理する方式。単件ごとに処理が完結するケースである。単件処理は要件により逐次又は並行で実行する。

[22] ここでは，業務アプリケーション（バッチ）のことを指す。

3.1.2.3.2 APレイヤ・コンポーネント定義

バッチ処理は，業務アプリケーション（バッチ）が機能を提供する。

バッチ処理方式における概念図及びAPレイヤ・コンポーネント構成を下図，下表に示す。なお，概念図中のアクセスパスは代表的なもののみ記載している。システム構成要素間のアクセスパスに関する説明は，「3.1.1.3.1.3 システム構成要素間のアクセスパス」を参照すること。

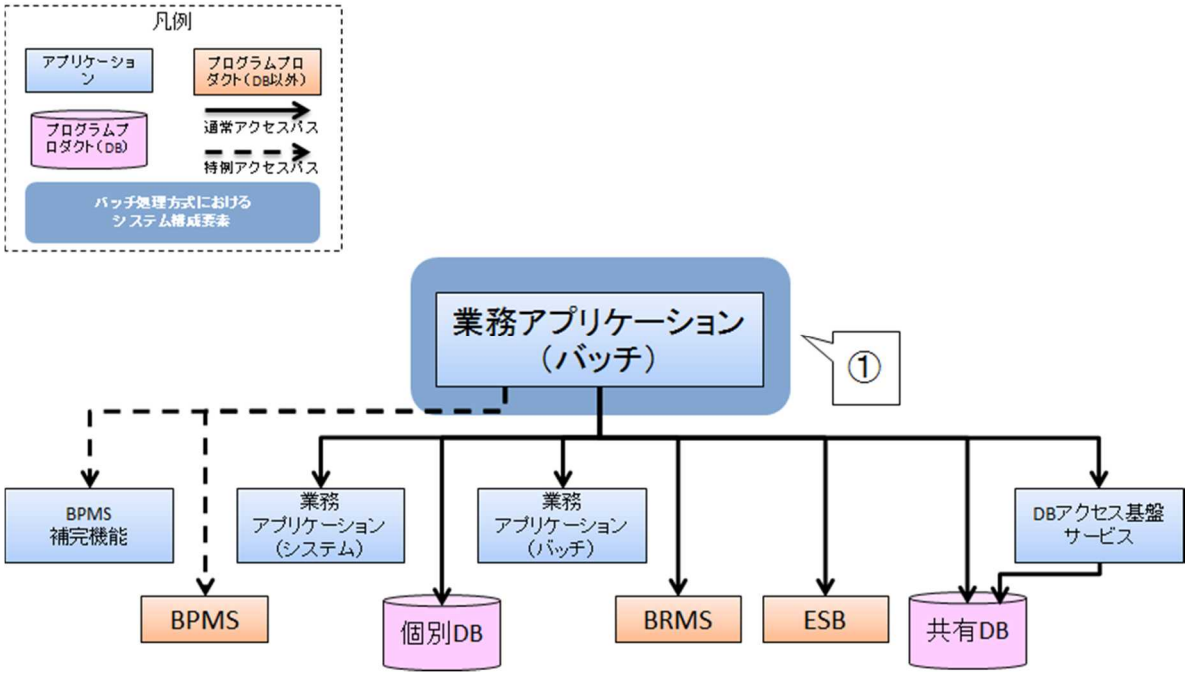
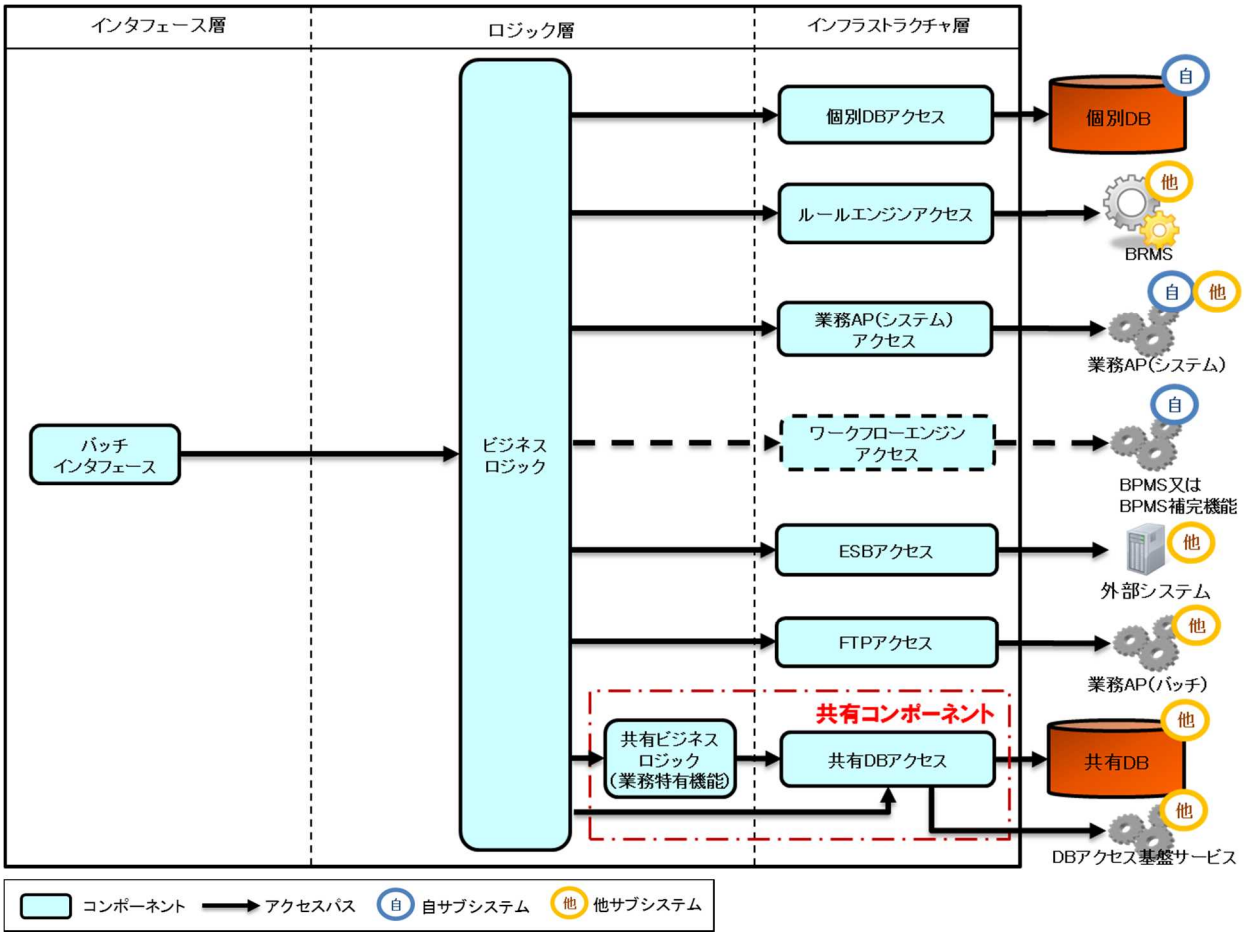


図 3.1-30 バッチ処理方式の概念図

① バッチ処理の内容に応じ，ビジネスロジックの実行のために他のシステム構成要素へアクセスする。

- 業務アプリケーション(バッチ)の場合



※「ワークフローエンジンアクセス」は，アクセスパスの特例において利用を許容する。
点線の矢印は，特例のアクセスパスを示している。
詳細は，「3.1.1.3.1.3 (4)アクセスパスの特例」を参照のこと。

図 3.1-31 APレイヤ・コンポーネント定義(バッチ)

表 3.1-29 コンポーネントの責務(バッチ)

項番	コンポーネント	配置するAPレイヤ	責務
1	バッチインタフェース	インタフェース層	インタフェースの入力項目チェック等，ビジネスロジックを呼び出す上での前提条件を満たすことを保証する。 また，実行結果を返却する。
2	ビジネスロジック	ロジック層	業務アプリケーション(バッチ)として提供する主たる機能を実装するコンポーネント。 必要に応じてインフラストラクチャ層のコンポーネントを呼び出し，参照・判定・更新といった処理の組み立てを行う。
3	共有ビジネスロジック (業務特有機能)	ロジック層	ビジネスロジックのうち，複数の業務アプリケーションで共有し，かつ特許庁業務特有の機能(期間計算等)を提供するコンポーネント。
4	共有DBアクセス	インフラストラクチャ層	共有データベースへの参照・更新機能を提供するコンポーネント。本コンポーネントは，特実記録原本管理システムの設計開発基準に記載されている「基盤API」を指す。詳細は『DBアクセス基盤利用者向けガイドライン － 6．基盤API』を参照のこと。
5	個別DBアクセス	インフラストラクチャ層	個別データベースへの参照・更新機能を提供するコンポーネント。ロジック層に対して汎用的なAPIを提供するために製品固有のデータアクセス技術，永続化手段を上位層から隠蔽する。 なお，横断的に利用する共通的な機能をコンポーネントとして提供し，データベースアクセス処理毎に開発が必要なSQL定義やエンティティクラス等はコンポーネントとは別に管理する。
6	ルールエンジン アクセス	インフラストラクチャ層	BRMSへルールの実行を要求する。 ビジネスルール管理サブシステムのサービス呼び出しを行う。 ロジック層に対してBRMSの呼び出し方式を隠蔽する。
7	業務AP(システム) アクセス	インフラストラクチャ層	業務アプリケーション(システム)が提供するサービスの呼び出しを行う。アクセスパスの制限事項については「3.1.1.3.1.3 (2)」参照。業務アプリケーション(システム)が提供するサービスを呼び出す際の通信技術や手順を上位層から隠蔽する。
8	ワークフローエンジン アクセス	インフラストラクチャ層	BPMS又はBPMS補完機能の呼び出しを行う。 ロジック層に対して汎用的なAPIを提供するために製品固有のAPIの呼び出しを上位層から隠蔽する。 なお，特異な条件時のアクセスとして許容される。詳細は，「3.1.1.3.1.3 (4)アクセスパスの特例」を参照のこと。
9	ESBアクセス	インフラストラクチャ層	外部システムの呼び出しを行う。外部システムを呼び出す際の通信技術や手順を上位層から隠蔽する。
10	FTPクライアント	インフラストラクチャ層	他サブシステムの業務アプリケーション(バッチ)に対し，ファイルでデータを連携するためにFTP通信を行う。FTP呼び出しを行う際の通信技術や手順を上位層から隠蔽する。

3.1.2.3.3 データの保護方式

バッチ処理方式のデータの保護方式は，オンライン処理方式のデータの保護方式と同様である。
詳細は「3.1.2.2.3 データの保護方式」を参照のこと。

3.1.2.3.4 エラー処理方式

バッチ処理方式で発生するエラーの種類を下表に示す。

表 3.1-30 エラーの種類

項番	エラーの種類	定義	例
1	業務エラー	業務設計(仕様)の中で想定しているエラー	● 入力ファイルの形式エラー ● データの相関エラー 等
2	システムエラー	業務設計(仕様)の中で想定していないエラー	● OSから通知されるエラー ● ミドルウェアから通知されるエラー 等

上記のエラーが発生した場合，以下の対応をするものとする。

表 3.1-31 エラー発生時の対応

項番	対応	目的
1	エラーの詳細をログ等に出力する	データの保護や，障害解析性を向上する。
2	データを整合性のある状態に戻す	データの整合性を確保する。 詳細は，「3.1.2.3.3 データの保護方式」を参照のこと。
3	システム監視サーバに通知して，監視端末にアラートを出す。	運用者にエラーを知らせる。

なお，システムエラーといった，業務設計(仕様)の中で想定されていないエラーを扱う処理は，通常の業務処理のプログラム中に混在して記述すると処理の流れの見通しが悪くなり，保守性が低下する。このような問題を回避するため，エラー処理を集約的に定義・実行できる仕組み(集約エラーハンドラ)を用いることとする。

また，各モジュールの責任範囲外の入力値にデータ形式の違い等があった場合，例外を発生させ，集約エラーハンドラで集約的にエラー処理を行うものとする。

3.1.2.3.5 業務継続方式

業務継続方式として，エラー発生時の再実行を，業務アプリケーション（バッチ）の中で行う場合と，業務アプリケーション（バッチ）の処理を一旦終了したのちジョブ管理システム等によって行う場合がある。

それぞれの業務継続方式の処理概要を下表に示す。

表 3.1-32 業務継続方式

項番	再実行の契機	業務継続方式	処理概要
1	業務アプリケーション（バッチ）	リトライ	エラー（例えばタイムアウト等）発生時，エラーとなった処理を自動的に再実行する。
2	ジョブ管理システム等	リスタート	エラー発生時には，コミット済みのリスタートポイントまでトランザクションをロールバックする。リカバリ時はリスタートポイントから処理を再実行する。
3		リラン	エラー発生時には，全てのトランザクションをロールバックする。リカバリ時は最初から処理を再実行する。

3.1.2.4 ディレードバッチ処理方式

目的:	ディレードバッチ処理方式のデータ保護方式，エラー処理方式，業務継続方式を各サブシステムで異なるものにならないよう定型化することにより，システムの信頼性の向上を図ると共に，システム構成要素（アプリケーション）の機能を画一化し，変更時の影響箇所の局所化を図る。
スコープ:	階層定型化サブシステム
指針1:	ディレードバッチ処理方式のデータ保護方式は，「3.1.2.4.3 データの保護方式」に従って設計すること。
指針2:	ディレードバッチ処理方式のエラー処理方式は，「3.1.2.4.4 エラー処理方式」に従って設計すること。
指針3:	ディレードバッチ処理方式の業務継続方式は，「3.1.2.4.5 業務継続方式」に示される業務継続方式のいずれかを採用した設計とすること。

以下，ディレードバッチ処理方式について処理方式パターンを示す。

3.1.2.4.1 処理方式パターン

ディレードバッチ処理方式における処理方式のパターンを下表に示す。

表 3.1-33 ディレードバッチ処理方式の処理方式パターン

項番	分類	説明
1	画面系	ユーザ操作により，画面からサーバに処理要求を行い，要求受領の旨の応答を即時に返却した後，要求された処理を行う方式。

3.1.2.4.2 APレイヤ・コンポーネント定義

ディレードバッチ処理は，画面からの処理要求に対して要求を受け付けた旨の応答を即時に返却した後，非同期に実際の処理を行う場合に用いる方式である。非同期処理の実現方式としては，ユーザからの処理要求を登録しユーザへ応答を返却するまでを行う部分と，ディレード処理要求管理情報を定期的に確認してバッチとして実行する部分とで構成する。前者の部分は「3.1.2.2 オンライン処理方式」で実現するものであり，本項にて特筆すべき点はない。ディレードバッチ処理方式としては，後者のディレード処理要求管理情報の確認から処理の実行までを行う部分を対象とする。ディレードバッチ処理は，業務アプリケーション（バッチ）が機能を提供する。

以下に概念図と処理概要を示す。なお，概念図中のアクセスパスは代表的なもののみ記載している。システム構成要素間のアクセスパスに関する説明は，「3.1.1.3.1.3 システム構成要素間のアクセスパス」を参照すること。

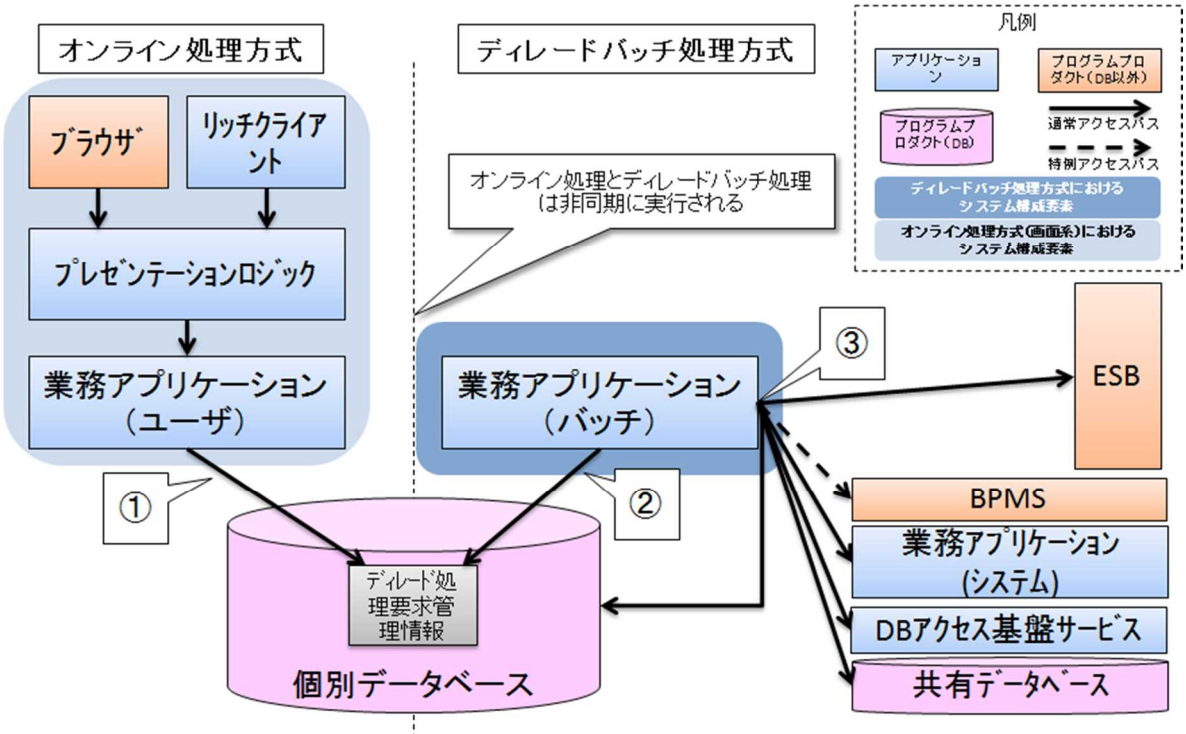


図 3.1-32 非同期処理とディレードバッチ処理方式の概念図

① 画面からの処理要求を「ディレード処理要求管理情報」として個別データベースに登録し，実際の処理の完了を待たずに要求を受け付けた旨の応答を画面へ返却するオンライン処理方式で実現する。それ以外にも，後続②以降のバッチ処理に対してインプットとなるデータを生成する等，要件に応じた処理を実行した上でディレード処理要求の登録を行ってもよい。

② ディレード処理の業務アプリケーション（バッチ）はあらかじめ起動済みで常駐プロセスとして動作し，定期的に「ディレード処理要求管理情報」を確認し処理要求を待ち受ける。登録された要求を確認すると要求内容に応じて③のバッチ処理を開始する。①の処理と②以降の処理は非同期に実行される。

③ 登録された処理要求の内容に応じたビジネスロジックを実行する。バッチ処理方式のビジネスロジック同様に，必要に応じて他のシステム構成要素にアクセスする。[23]

[23] ディレードバッチ処理を契機にビジネスプロセスを進める方法として，プレゼンテーションロジックからBPMSへアクセスする方法（方法1）と，業務アプリケーション（バッチ）からBPMSへアクセスする方法（方法2）がある。原則，ビジネスプロセスの中のタスクを同期させる場合は方法1，同期させる必要がない場合は方法2を採用する。ただし，方法1ではBPMN上に待機するフローを記載する必要があり，それがビジネスプロセスの可視化の点で妥当でない場合は，方法2での対応を考慮する。

「ディレード処理要求管理情報」の閲覧や再実行のための更新といった機能は，業務要件や運用要件によりユーザ向けの画面として実装したり運用ツールとして実装することを想定している。

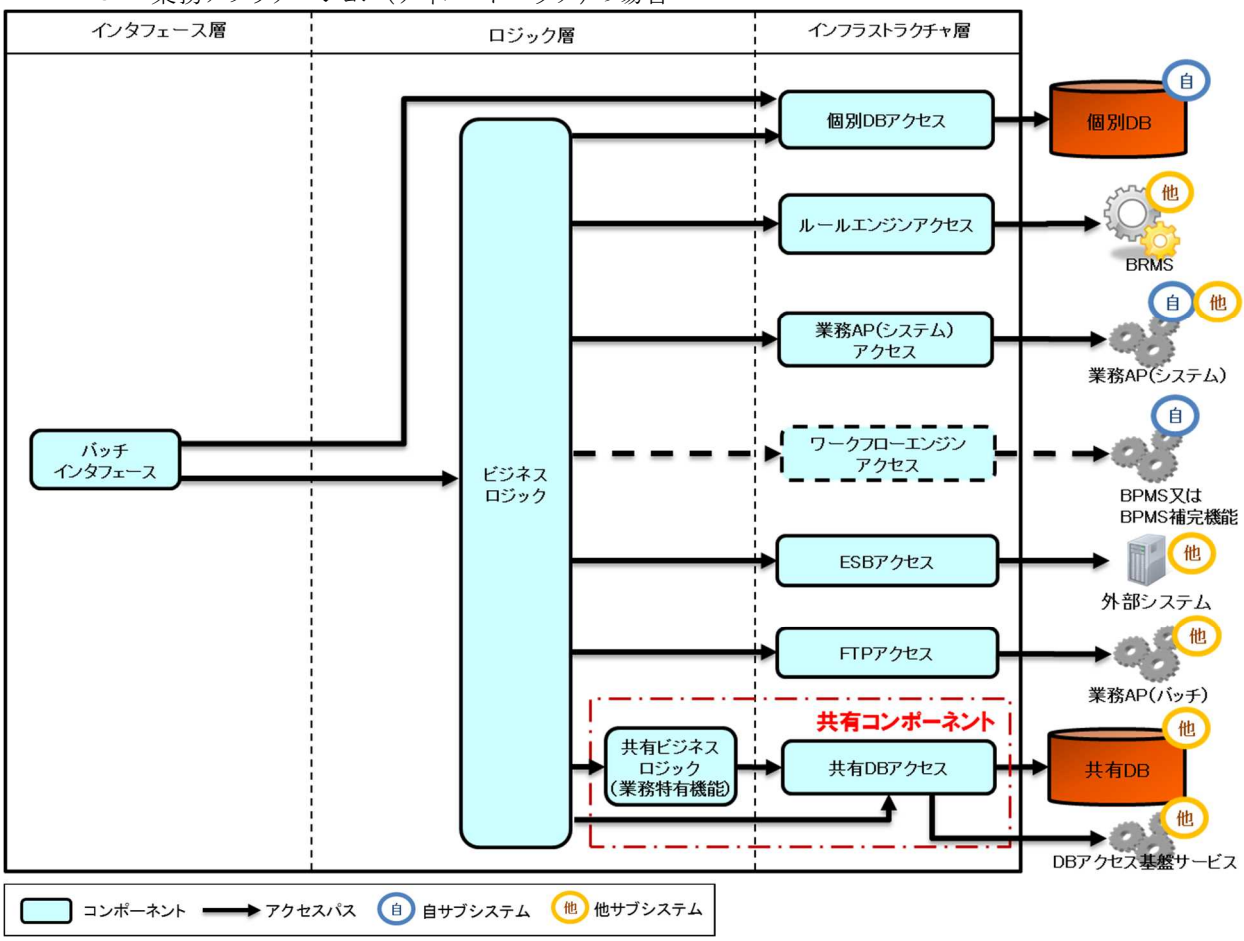
ディレードバッチ処理の業務継続方式（詳細は後述の「3.1.2.4.5 業務継続方式」を参照すること）を実現するために「ディレード処理要求管理情報」には以下に示す管理情報に相当する項目を保持する。

表 3.1-34 ディレード処理要求管理情報の一例

項番	管理項目	説明
1	識別情報	ディレード処理要求を識別するための情報を管理する。
2	ディレード処理種別	実行するビジネスロジックとの対応付けのための情報を管理する。
3	入力情報	ディレード処理のビジネスロジックにインプットとして渡すデータやファイルパス等の情報を管理する。
4	ディレード処理結果	ディレード処理の処理結果を管理する。
5	ディレード処理状況	ディレード処理の状況（未実行，実行中，完了 等）を管理する。
6	上記以外の管理項目	処理開始／終了時刻等，必要に応じ管理する。

ディレードバッチ処理方式におけるAPレイヤ・コンポーネント構成を下図，下表に示す。バッチインタフェース コンポーネントの責務とアクセスパス以外は，「3.1.2.3.2 APレイヤ・コンポーネント定義」と同様である。

- 業務アプリケーション（ディレードバッチ）の場合



※「ワークフローエンジンアクセス」は，アクセスパスの特例において利用を許容する。
点線の矢印は，特例のアクセスパスを示している。
詳細は，「3.1.1.3.1.3 (4)アクセスパスの特例」を参照のこと。

図 3.1-33 APレイヤ・コンポーネント定義（ディレードバッチ）

表 3.1-35 コンポーネントの責務(ディレードバッチ)

項番	コンポーネント	配置するAPレイヤ	責務
1	バッチインタフェース	インタフェース層	個別データベースの「ディレード処理要求管理情報」を定期的に確認し，処理要求の登録を確認すると要求内容に応じたビジネスロジックを実行する。

※上記以外のコンポーネントについては「表 3.1-29 コンポーネントの責務(バッチ)」と同様である。

3.1.2.4.3 データの保護方式

ディレードバッチ処理方式のデータの保護方式は，オンライン処理方式のデータの保護方式と同様である。

詳細は「3.1.2.2.3 データの保護方式」を参照のこと。

3.1.2.4.4 エラー処理方式

ディレードバッチ処理で発生するエラーの種類を下表に示す。

表 3.1-36 エラーの種類

項番	エラーの種類	定義	例
1	業務エラー	業務設計(仕様)の中で想定しているエラー	● 入力ファイルの形式エラー ● データの相関エラー 等
2	システムエラー	業務設計(仕様)の中で想定していないエラー	● OSから通知されるエラー ● ミドルウェアから通知されるエラー 等

上記のエラーが発生した場合，以下の対応をするものとする。

表 3.1-37 エラー発生時の対応

項番	対応	目的
1	エラーの詳細をログ等に出力する	データの保護や，障害解析性を向上する。
2	データを整合性のある状態に戻す	データの整合性を確保する。 詳細は，「3.1.2.3.3 データの保護方式」を参照のこと。
3	エラーの発生を「ディレード処理要求管理情報」のディレードバッチ処理結果に反映する	システム利用者や運用者にエラーを知らせる。

なお，システムエラーといった，業務設計(仕様)の中で想定されていないエラーを扱う処理は，通常の業務処理のプログラム中に混在して記述すると処理の流れの見通しが悪くなり，保守性が低下する。このような問題を回避するため，エラー処理を集約的に定義・実行できる仕組み(集約エラーハンドラ)を用いることとする。

また，各モジュールの責任範囲外の入力値にデータ形式の違い等があった場合，例外を発生させ，集約エラーハンドラで集約的にエラー処理を行うものとする。

3.1.2.4.5 業務継続方式

ディレードバッチの業務継続方式では，エラー発生箇所がバッチインタフェース（ディレード処理要求の制御）の場合とそれ以外の場合で異なる。

それぞれの業務継続方式の処理概要を下表に示す。

表 3.1-38 業務継続方式

項番	再実行の契機	エラー発生箇所	業務継続方式	処理概要
1	業務アプリケーション（バッチ）	バッチインタフェース（ディレード処理要求の制御）	再起動	エラー発生時には，実行中のトランザクションをロールバックする。リカバリ時は最初から処理を再実行する。 （再起動では「ディレード処理要求管理情報」に再実行対象が格納されていないため，再実行対象を設定した上で処理を継続する）
2		バッチインタフェース（ディレード処理要求の制御）以外	リラン	エラー発生時には，全てのトランザクションをロールバックする。リカバリ時は最初から処理を再実行する。 （リランでは「ディレード処理要求管理情報」の再実行対象の情報に基づき処理を継続する）
3			リスタート	エラー発生時には，コミット済みのリスタートポイントまでトランザクションをロールバックする。リカバリ時はリスタートポイントから処理を再実行する。
4			リトライ	エラー（例えばタイムアウト等）発生時，エラーとなった処理を自動的に再実行する。

エラー発生箇所がバッチインタフェース（ディレード処理要求の制御）（項番1）の場合，「ディレード処理要求管理情報」のディレード処理結果やディレード処理状況の更新，ディレード処理要求の制御のための参照・更新でのエラーになるため，業務アプリケーション（バッチ）の再起動により業務を継続する。

エラー発生箇所がバッチインタフェース（ディレード処理要求の制御）以外（項番2～4）[24]の場合，リラン及びリスタートの契機は，「ディレード処理要求管理情報」のディレード処理結果やディレード処理状況の更新により，ディレード処理要求の制御部分に再実行対象として認識させることで実現する。

[24]業務継続方式がリトライの場合は，リトライによってエラーを復旧できなかった場合

3.1.2.5 帳票出力方式

現状，各個別業務システム内に配置されている帳票出力機能は，ToBeアーキテクチャにおいて紙出力サブシステムへ集約化される想定である。紙出力サブシステムの位置付けは，以下のとおり。

- 帳票出力機能を変更する際の影響範囲の局所化を狙い，紙出力共通のサブシステムとして，紙出力部分の一本化を図る。
- 帳票用プロダクトを利用した専用の作りが必要となるため，インタフェース定型化システムとして扱う(「表 2.1-1 インタフェース定型化サブシステムの例」)。

上記のとおり，帳票出力機能は個別のアーキテクチャ(インタフェース定型化システム)として定めることとなるため，「2.1 スコープ(ルールの適用範囲)」に示される内容に従い，帳票出力方式については本書では扱わないものとする。

3.1.2.6 サブシステム間連携処理方式

目的:	サブシステム間の連携方法を統一することにより，各サブシステムでの連携時の処理がパターン化され，サブシステム構造の定型化を促進する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
指針1:	BPMSを介したサブシステム間連携方式は，「3.1.2.6.1.1 (1)連携方式」，「3.1.2.6.1.1 (2)設計・開発時の考慮点」に従って設計すること。
指針2:	BPMSとジョブ管理システム間の連携方式は，「3.1.2.6.1.2 (1)連携方式」，「3.1.2.6.1.2 (2)設計・開発時の考慮点」に従って設計すること。
指針3:	バッチ処理方式のサブシステム間連携方式は，「3.1.2.6.1.3 (1)連携方式」，「3.1.2.6.1.3 (2)設計・開発時の考慮点」に従って設計すること。
指針4:	外部システム連携層を介した外部システム連携方式は，「3.1.2.6.1.4 (1)連携方式」，「3.1.2.6.1.4 (2)設計・開発時の考慮点」，「3.1.2.6.1.4 (4)ESBに関する特記事項」及び「3.1.2.6.1.4 (5)外部システム互換機能に関する特記事項」に従って設計すること。

目的:	サブシステム間連携処理方式のデータ保護方式，業務継続方式を各サブシステムで異なるものにならないよう定型化することにより，システムの信頼性の向上を図ると共に，システム構成要素（アプリケーション）の機能を画一化し，変更時の影響箇所の局所化を図る。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
指針1:	サブシステム間連携処理方式のデータ保護方式は，「3.1.2.6.2 データの保護方式」に従って設計すること。
指針2:	サブシステム間連携処理方式の業務継続方式は，「3.1.2.6.3 業務継続方式」に従って設計すること。

サブシステム間連携処理方式とは，「3.1.1.3.1.2 処理方式に基づくシステム構成要素」において示した処理方式同士の連携や外部システム連携層を介した内部システムと外部システムとの連携に関する処理方式を意味する。以下，連携処理方式について処理方式パターンを示す。

3.1.2.6.1 処理方式パターン

サブシステム間連携処理方式における処理方式のパターンを下表に示す。

表 3.1-39 連携処理方式の処理方式パターン

項番	分類	組み合わせ	説明
1	BPMSを介したサブシステム間連携	オンライン処理方式とオンライン処理方式の連携 ※サービス系の連携	業務フローとしてサブシステム間の連携を行う場合は，その連携ルートをBPMS上のワークフローにて記述し，BPMSを介して他サブシステムのBPMSや業務アプリケーション（システム）と連携を行う方式。 詳細は「3.1.2.6.1.1 BPMSを介したサブシステム間連携」を参照のこと。
2	BPMSとジョブ管理システム間の連携	オンライン処理方式とバッチ処理方式の連携	ToBeアーキテクチャでは，基本的にBPMSによって単件オンライン化されるが，一部の業務では業務アプリケーション（バッチ）により一括で処理を行うものも存在する。それらの処理を連携する処理方式。 オンライン処理方式（サービス系）とバッチ処理方式を組み合わせて，データベースやファイルでデータの授受を行い，スケジューラ起動等の契機により連携処理を行う。 詳細は「3.1.2.6.1.2 BPMSとジョブ管理システム間の連携」を参照のこと。

項番	分類	組み合わせ	説明
3	バッチ処理方式のサブシステム間連携	バッチ処理方式とバッチ処理方式の連携	サブシステム間にて，バッチ処理の連動やデータの授受を行う方式。 共有データベースやFTP(S)によるデータ授受を行い，ジョブ管理システムを介して連携を行う。 詳細は「3.1.2.6.1.3 バッチ処理方式のサブシステム間連携」を参照のこと。
4	外部システム連携層を介した外部システム連携	内部システムと外部システムの連携	外部システムとの連携には，それぞれの外部システムに固有のインタフェースやプロトコルが存在するため，それらのギャップを一元的に吸収する外部システム連携層を介してのみ行うものとする。その際の処理方式。 内部システムから外部システム連携を行う際，外部システム連携層が提供する内部インタフェースと同等のアクセス方法を用いて処理を実施する。 詳細は「3.1.2.6.1.4 外部システム連携層を介した外部システム連携」を参照のこと。

なお，分散ファイルシステム（ファイル共有，NFS）によるサブシステム間連携処理方式は，アプリケーションの疎結合化・変更影響の局所化の観点から，採用しない。理由は，分散ファイルシステムが，データがどのシステム上に存在するかを意識することなくアクセスでき，読み込み・書き込み箇所がサブシステムを跨って共有される性質のものであるため，FTPを用いた連携処理方式よりもサブシステム間の関係が密結合になり，依存関係が強まるからである。

3.1.2.6.1.1 BPMSを介したサブシステム間連携

(1) 連携方式

BPMSを介したサブシステム間連携の連携方式を下表に示す。サブシステムのワークフローが，他のサブシステムとオンラインで連携する場合は，下表のいずれかの方式を利用すること。

表 3.1-40 BPMSを介したサブシステム間連携の連携方式

項番	連携方式	概要
1	ワークフロー連携	BPMSから他サブシステムのBPMSにアクセスし，開始又は待機中のワークフローを再開する。(ワークフロー呼出) ただし，待機可能な箇所が複数あるワークフローを再開する場合，現在どこで待機しているか判断する必要があるため，一旦BPMS補完機能にアクセスし，BPMS補完機能が待機場所を特定しワークフローを再開する[25]。(BPMS補完機能呼出) BPMS補完機能呼出の処理方式は，「オンライン処理方式」の「サービス系」と同様である。 詳細は，「3.1.2.2 オンライン処理方式」を参照のこと。
2	サービス連携	BPMS又はBPMS補完機能から他サブシステムの業務アプリケーション(システム)にアクセスし，連携データの受け渡し等を行う。 なお，処理方式は，「オンライン処理方式」の「サービス系」と同様である。 詳細は，「3.1.2.2 オンライン処理方式」を参照のこと。

(2) 設計・開発時の考慮点

設計・開発時に考慮すべき点を以下に示す。

- BPMSのプロセスデータ[26]は，連携時に引数で受け渡しを行うこと。
- BPMSのプロセスデータ以外のデータの受け渡しを行う場合は，共有データベースを用いること。
- 共有データベースの特定サブシステム間共有データ[27]に一時的に保持される受け渡しデータは，データを受領するサブシステム側で使用後に削除すること。

[25] 具体的な内容は，「3.3.2.2.2 (1) 起動するビジネスプロセスやイベント通知するビジネスプロセスインスタンス及びビジネスプロセス上の対象イベントを業務的な条件で動的に特定する処理」を参照のこと。

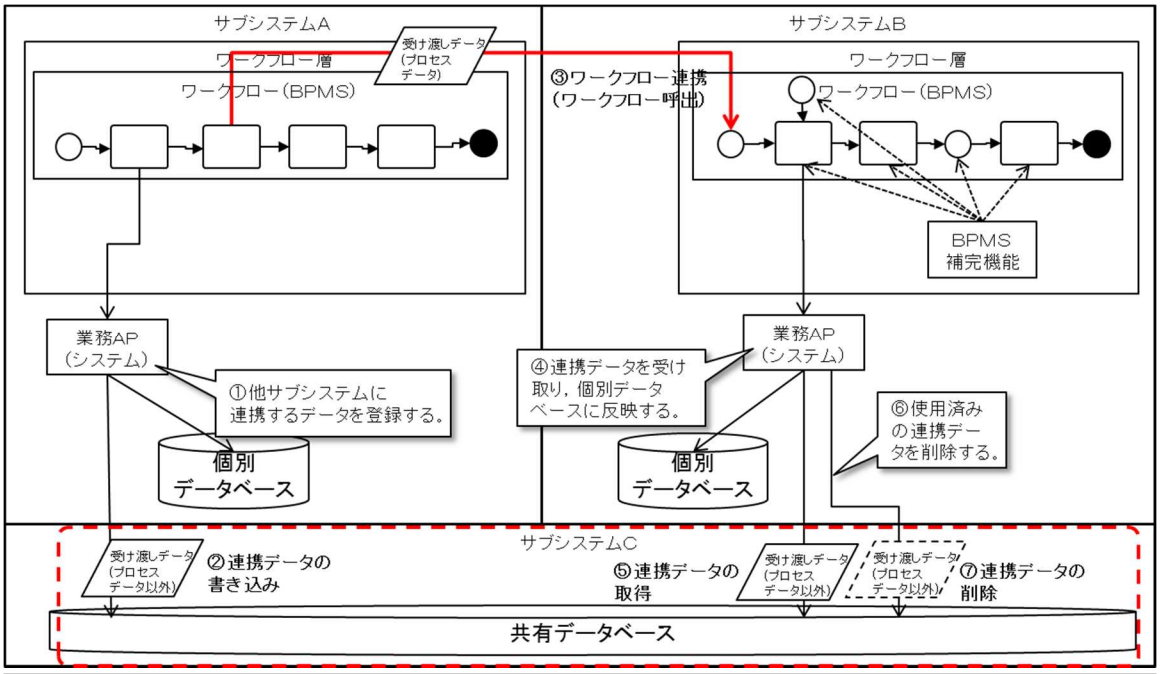
[26] 「3.3.2.1.3 プロセスデータとして保持する情報」を参照のこと。

[27] 詳細は，『データ統合方針書』の「2.3 データ配置の考え方」を参照のこと。

(3)　処理方式のイメージ

サブシステム間連携における処理方式のイメージを以下に示す。

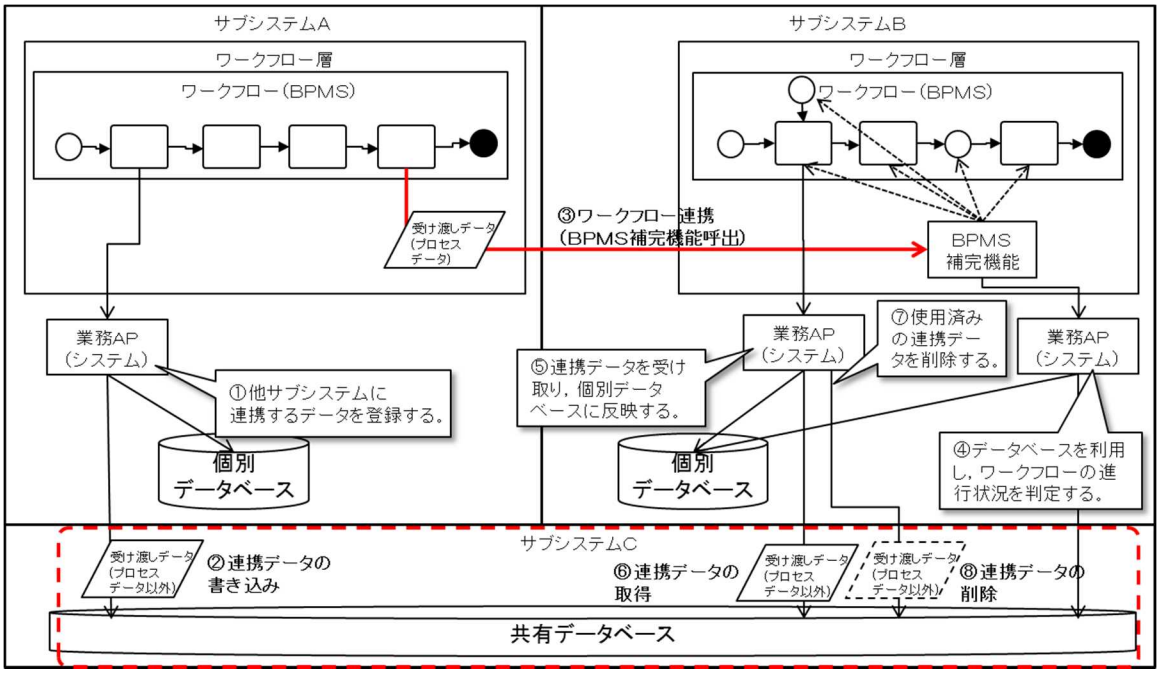
- ワークフロー呼び出しによるワークフロー連携処理方式



- BPMS のプロセスデータ以外の連携先のサブシステムとのデータの受け渡しは共有データベースを使用する(①, ②)。
- ワークフロー(BPMS)を介したサブシステムの連携を行う。BPMS のプロセスデータは，連携時に引数で受け渡す。ワークフローから連携先サブシステムのワークフローにアクセスする(③)。
- 連携先のサブシステムでは，共有データベース又は引数で受け取ったデータに基づき処理を行う(④, ⑤)。
- 連携先のサブシステムにて，使用済みの連携データを削除する(⑥, ⑦)。

図 3.1-34 ワークフロー呼び出しによるワークフロー連携処理イメージ

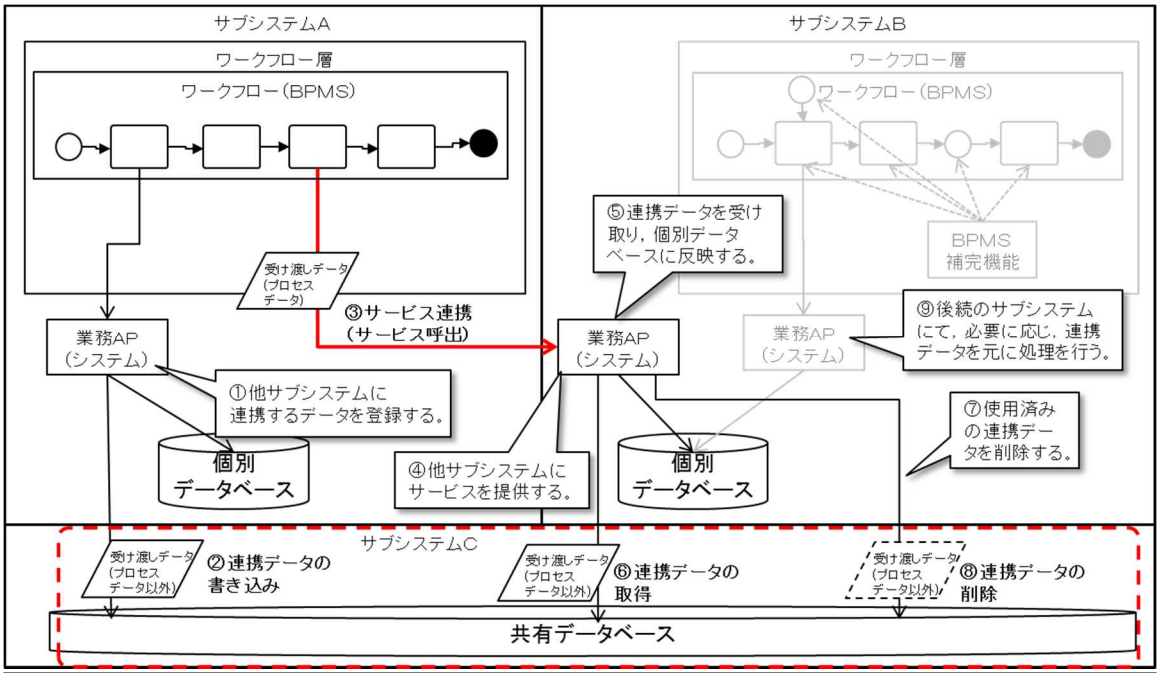
- BPMS補完機能呼び出しによるワークフロー連携処理方式



- BPMS のプロセスデータ以外の連携先のサブシステムとのデータの受け渡しは共有データベースを使用する(①, ②)。
- ワークフロー(BPMS)を介したサブシステムの連携を行う。BPMS のプロセスデータは，連携時に引数で受け渡す。待機可能な箇所が複数ある連携先サブシステムのワークフローを再開する場合，BPMS 補完機能を呼び出し，BPMS 補完機能がワークフロー再開の判断を行う(③)。
- ワークフロー再開の判断に必要な情報の取得を業務 AP(システム)に依頼する(④)。
- 連携先のサブシステムでは，共有データベース又は引数で受け取ったデータに基づき処理を行う(⑤, ⑥)。
- 連携先のサブシステムにて，使用済みの連携データを削除する(⑦, ⑧)。

図 3.1-35 BPMS補完機能呼び出しによるワークフロー連携処理イメージ

- サービス呼び出しによるサブシステム間連携処理方式



- BPMS のプロセスデータ以外の連携先のサブシステムとのデータの受け渡しは共有データベースを使用する(①, ②)。
- ワークフロー(BPMS)を介したサブシステムの連携を行う。BPMS のプロセスデータは，連携時に引数で受け渡す。ワークフローから連携先サブシステムのサービス(業務 AP(システム))にアクセスする(③, ④)。
- 連携先のサブシステムでは，共有データベース又は引数で受け取ったデータに基づき処理を行う(⑤, ⑥)。
- 連携先のサブシステムにて，使用済みの連携データを削除する(⑦, ⑧)。
- 連携先のサブシステムにおいて，必要に応じて，個別データベースに格納された連携データを元に処理を行う(⑨)。

図 3.1-36 サービス呼び出しによるサブシステム間連携処理イメージ

3.1.2.6.1.2 BPMSとジョブ管理システム間の連携

(1) 連携方式

BPMSとジョブ管理システム間の連携方式を下表に示す。BPMSとジョブ管理システムの間で連携する場合において，BPMSからジョブ管理システムへ連携する場合は下表の項番1の方式を，ジョブ管理システムからBPMSに連携する場合は下表の項番2の方式を利用すること。

表 3.1-41 BPMSとジョブ管理システム間の連携方式

項番	連携方式	概要
1	BPMSからジョブ管理システムに連携する。	BPMS（単件処理）からジョブ管理システム（一括処理）に連携して処理を行う場合，以下のように処理を行う。 ① BPMSは，業務アプリケーション（システム）を経由して，単件処理毎に受け渡しデータを共有データベースに格納する。 ② ジョブ管理システムは，時間起動等のタイミングで，業務アプリケーション（バッチ）を実行する。 ③ 業務アプリケーション（バッチ）は，共有データベースから複数の受け渡しデータを抽出し，一括で処理を行う。
2	ジョブ管理システムからBPMSに連携する。	ジョブ管理システム（一括処理）からBPMS（単件処理）に連携して処理を行う場合，以下のように処理を行う。 ① ジョブ管理システムは，時間起動等のタイミングで，業務アプリケーション（バッチ）を実行する。 ② 連携先サブシステムの業務アプリケーション（バッチ）は，共有データベースから複数の受け渡しデータを抽出し，単件処理に分割してビジネスプロセスを開始する。この際，BPMSに受け渡しデータのプロセスデータを引数として受け渡す。 ③ BPMSは，業務アプリケーション（システム）を経由して，共有データベースからプロセスデータを条件に受け渡しデータを抽出し，単件処理を行う。

(2) 設計・開発時の考慮点

設計・開発時に考慮すべき点を下表に示す。

表 3.1-42 設計・開発時の考慮点

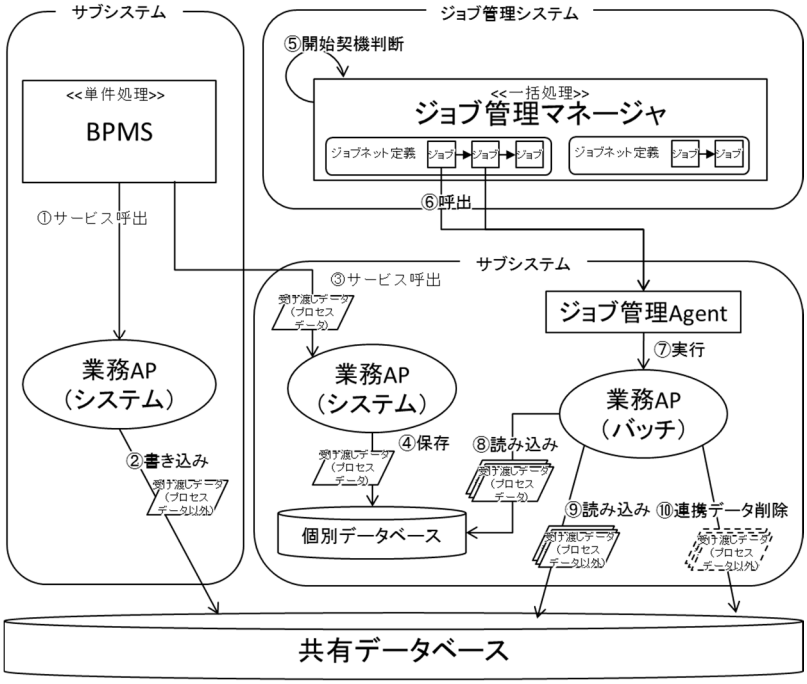
項番	考慮点		説明
1	BPMSからジョブ管理システムに連携する場合	データの受け渡し	共有データベースを用いてデータの受け渡しを行うこと。 BPMSのプロセスデータ[26]は，BPMSからジョブ管理システムが実行する業務アプリケーション（バッチ）と同一のサブシステムである業務アプリケーション（システム）へ引数で渡し，個別データベースに格納すること。（BPMSから渡された引数（プロセスデータ）はバッチ処理実行時まで個別データベースに格納しておく。） バッチ処理実行後に不要となった受け渡しデータ（共有データベースに格納されたプロセスデータ以外のデータ及び個別データベースに格納されたプロセスデータ）は，データを受領するサブシステムが削除すること。
2		BPMS終了時点の振る舞い	ジョブ管理システム側が時間起動（日次等）であれば，単件処理の終了時点でジョブ管理システムと連携する必要はなく，共有データベース及び／又は個別データベースに連携用データを格納するだけでよい。
3		ジョブ管理システム（一括処理）の開始契機	時間起動であればジョブ管理システムが自ら契機を判断する。 連携元サブシステムからバッチを起動する契機を与える場合は，以下のいずれかをジョブ管理システムが検知することにより実現すること。 ● 連携元サブシステムからの要求により連携先サブシステム[28]に格納又は更新されたデータ ● 連携元サブシステムの要求により連携先サブシステムに格納された検知用のファイル
4	ジョブ管理システムからBPMSに連携する場合	データの受け渡し	共有データベース又はFTP(S)を用いてデータの受け渡しを行うこと。また，以下の点を考慮すること。 ● 共有データベースで受け渡した連携データは，データを受領するサブシステム側が使用後に削除すること。 ● FTPで送信したファイルは，ファイルを送信するサブシステム側が削除すること。 ● FTPで受信したファイルは，ファイルを受信するサブシステム側が使用後に削除すること。 以下に示すケースに適合する場合は，FTP(S)を使用しファイルにてデータを受け渡すこと。 ● 大容量ファイルを効率よく転送したい場合や一括処理のため複数件数のレコードデータを一つのファイルにまとめてやり取りしたい場合等，データベース経由では，効率が極めて悪い場合。
5		ジョブ管理システム（一括処理）の終了時点の振る舞い	連携元のバッチ処理の終了を契機に，連携先のバッチ処理を実行すること。BPMSは単件単位で実行する必要があるため，連携先の業務AP（バッチ）にて共有データベースに登録されている複数件のデータを単件単位に分解し，プロセスデータとしてBPMSへ引き渡し，ビジネスプロセスを開始すること。
6		BPMS（単件処理）の開始契機	「ジョブ管理システム（一括処理）の終了時点の振る舞い」に記載のタイミングでBPMSを開始すること。

[28] ジョブ管理システムが実行する業務アプリケーション（バッチ）を持つサブシステムのこと。

(3) 処理方式のイメージ

BPMSとジョブ管理システム間における処理方式のイメージを下図に示す。

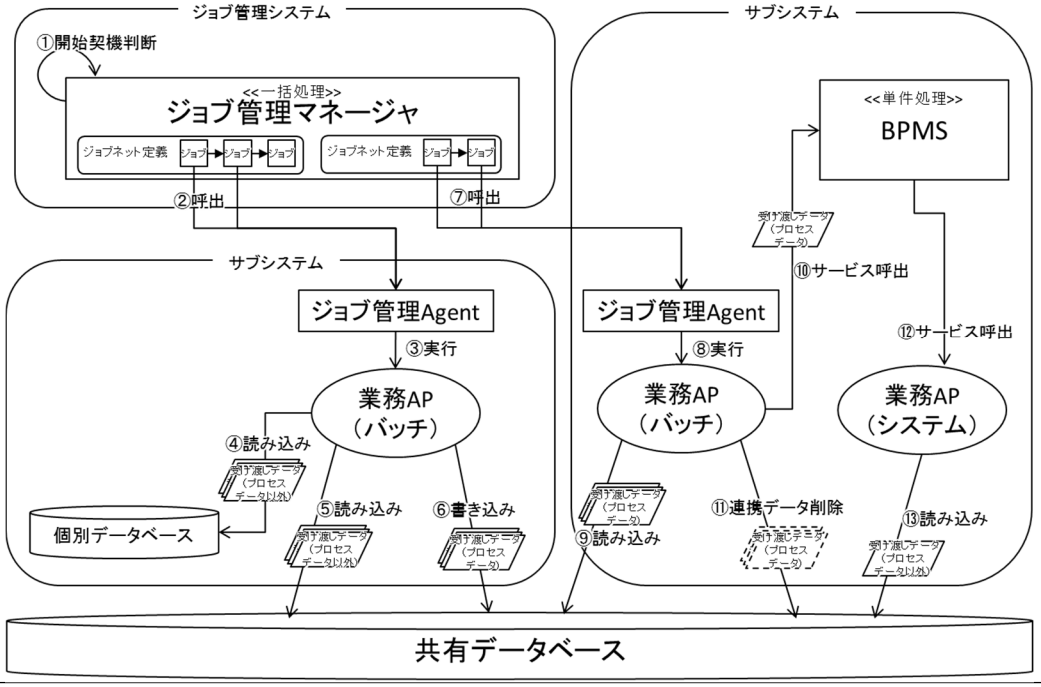
- BPMS → ジョブ管理システム連携方式



- BPMSのプロセスデータ以外の連携先のサブシステムとのデータの受け渡しは共有データベースを使用する(①, ②)。
- BPMSのプロセスデータは, BPMSから業務AP(バッチ)と同一のサブシステムである業務AP(システム)が受け取り, 引数で渡されたデータを保存する(③, ④)。
- ジョブの実行時に, 上記で保存したデータと共有データベースに格納されたデータを読み込み, 一括処理を行う(⑤～⑨)。
- 連携先のサブシステムにて, 使用済みの連携データを削除する(⑩)。

図 3.1-37 BPMSとジョブ管理システム間連携の処理イメージ(BPMS → ジョブ管理システム連携)

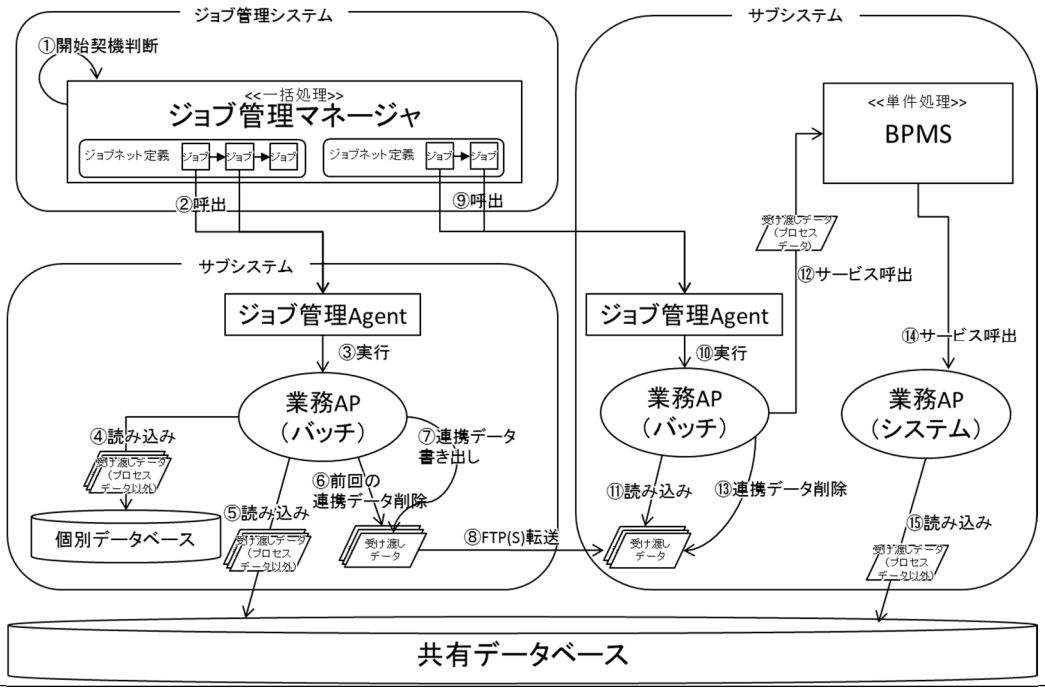
- ジョブ管理システム → BPMS連携方式(共有データベース経由)



- 連携先のサブシステムとのデータの受け渡しは共有データベースを使用する(①～⑥)。
- 連携元のジョブネットが終了し, 連携先のジョブネットが開始される(⑦, ⑧)。
- 連携先の業務AP(バッチ)は, データベースから複数件のデータを取得し, 単件単位にBPMSを呼び出す(⑨, ⑩)。
- 連携先のサブシステムにて, 使用済みの連携データを削除する(⑪)。
- BPMSは, 業務AP(システム)経由で, 共有データベースからプロセスデータを条件に受け渡しデータを取得し, 単件処理を行う(⑫, ⑬)。

図 3.1-38 BPMSとジョブ管理システム間連携の処理イメージ(ジョブ管理システム → BPMS連携(DB))

- ジョブ管理システム → BPMS連携方式(FTP(S)経由)



- 連携先のサブシステムとのデータの受け渡しはFTP(S)によるファイル転送で行う(①～⑧)。図中ではファイル転送が失敗した場合の再送に備え，連携データの作成前のタイミング(⑦)で，前回の連携データを削除している(⑥)。
- 連携元のジョブネットが終了し，連携先のジョブネットが開始される(⑨，⑩)。
- 連携先の業務AP(バッチ)は，転送されたファイルから複数件のデータを取得し，単件単位にBPMSを呼び出す(⑪，⑫)。
- 連携先のサブシステムにて，使用済みの連携データを削除する(⑬)。
- BPMSは，業務AP(システム)経由で，共有データベースからプロセスデータを条件に受け渡しデータを取得し，単件処理を行う(⑭，⑮)。

図 3.1-39 BPMSとジョブ管理システム間連携の処理イメージ(ジョブ管理システム → BPMS連携(FTP))

3.1.2.6.1.3 バッチ処理方式のサブシステム間連携

(1) 連携方式

バッチ処理方式の連携は,『運用管理エージェント登録ガイドライン』に準じて，ジョブ管理システムで管理されるジョブネットを利用する。ジョブネットは，サブシステム毎に構成するため，他サブシステムのバッチ処理と連携を行う場合には，以下に示す方法で連携を行う。

- 他サブシステムのジョブネットとの連携は，時刻起動あるいはFTP(S)等によるファイル到着契機で実行すること。

(2) 設計・開発時の考慮点

設計・開発時に考慮すべき点を以下に示す。

- データの受け渡しは，共有データベース又はFTP(S)を用いて行うこと。
- 以下に示す「FTP(S)の使用条件」に適合する場合は，FTP(S)を使用しファイルにてデータを受け渡すこと。

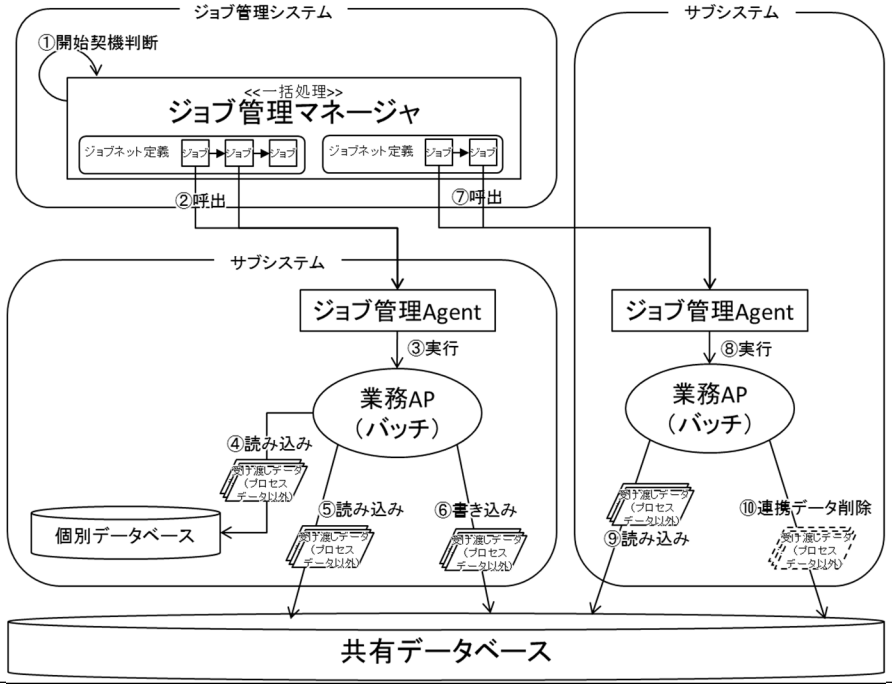
  FTP(S)の使用条件：
  - 大容量ファイルを効率よく転送したい場合や一括処理のため複数件数のレコードデータを一つのファイルにまとめてやり取りしたい場合等，データベース経由では，効率が極めて悪い場合。
- FTP(S)で送信したファイルは，ファイルを送信するサブシステム側が削除すること。
- 共有データベースの特定サブシステム間共有データ[29]に一時的に保持される受け渡しデータやFTP(S)で受領したファイルは，データを受領するサブシステム側で使用後に削除すること。

[29] 詳細は,『データ統合方針書』の「2.3 データ配置の考え方」を参照のこと。

(3)　処理方式のイメージ

バッチ処理によるサブシステム間連携方式のイメージを下図に示す。

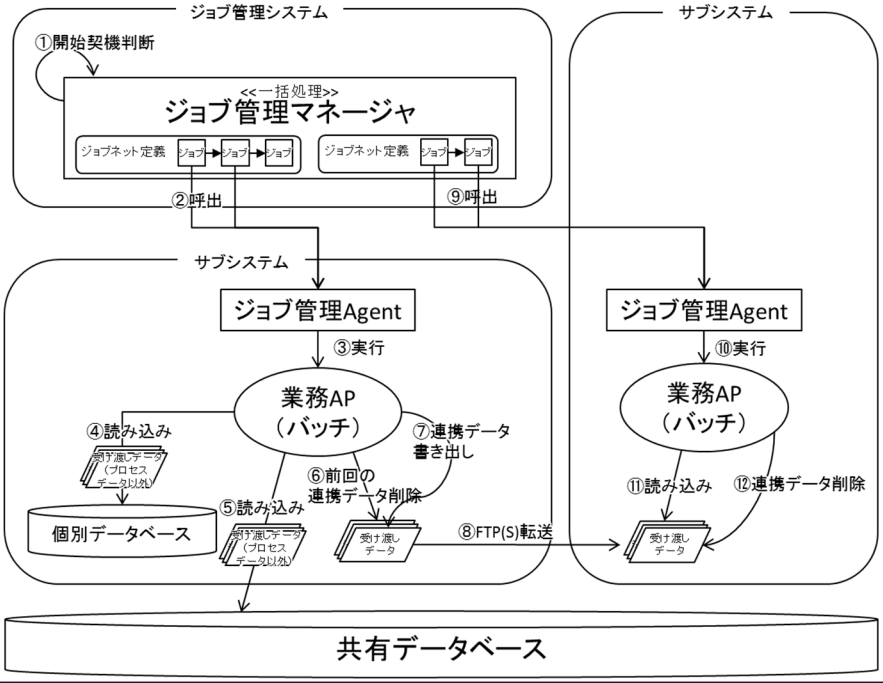
- バッチ処理による連携方式(共有データベース経由)



- 連携先のサブシステムとのデータの受け渡しは共有データベースを使用する(①～⑥)。
- 連携元のジョブネットが終了し，連携先のジョブネットが開始される(⑦，⑧)。
- 連携先のサブシステムでは，共有データベース経由で受け取ったデータに基づき処理を行う(⑨)。
- 連携先のサブシステムにて，使用済みの連携データを削除する(⑪)。

図 3.1-40 バッチ処理による連携処理イメージ(共有データベース経由)

- バッチ処理による連携方式(FTP(S)経由)



- 連携先のサブシステムとのデータの受け渡しはFTP(S)によるファイル転送で行う(①～⑧)。図中ではファイル転送が失敗した場合の再送に備え，連携データの作成前のタイミング(⑦)で，前回の連携データを削除している(⑥)。
- 連携元のジョブネットが終了し，連携先のジョブネットが開始される(⑨，⑩)。
- 連携先のサブシステムでは，FTP(S)経由で受け取ったデータに基づき処理を行う(⑪)。
- 連携先のサブシステムにて，使用済みの連携データを削除する(⑫)。

図 3.1-41 バッチ処理による連携処理イメージ(FTP(S)経由)

3.1.2.6.1.4 外部システム連携層を介した外部システム連携

(1) 連携方式

外部システムとの連携は，外部システム連携層を介して連携すること。

外部システム連携層の責務を実現する外部システム連携サブシステムは，インタフェース定型化サブシステムのため，外部システム連携層に持たせる機能等は，外部システム連携サブシステムの設計に関与するステークホルダ（設計・開発ベンダ等）が個別にアーキテクチャを定めるものとする。

外部システム連携層に持たせる機能は，「3.3.4.1.2 外部システム連携層に持たせる機能」を参照のこと。また，内部システムとの通信におけるプロトコルは，「3.2.1.1.2 内部インタフェースのプロトコル」を参照のこと。

(2) 設計・開発時の考慮点

外部システム連携層を介した外部システム連携のうち，階層定型化サブシステムと外部システム連携層（インタフェース定型化サブシステム）の通信において考慮すべき点を以下に示す。

- BPMSのプロセスデータは，連携時に引数で受け渡しを行うこと。
- BPMSのプロセスデータ以外のデータの受け渡しを行う場合は，共有データベースを用いること。
- 以下に示す「FTP(S)の使用条件」に適合する場合は，FTP(S)を使用しファイルにてデータを受け渡すこと。

  FTP(S)の使用条件：
  - 大容量ファイルを効率よく転送したい場合や一括処理のため複数件数のレコードデータを一つのファイルにまとめてやり取りしたい場合等，データベース経由では，効率が極めて悪い場合。
- 共有データベースの特定サブシステム間共有データ[30]に一時的に保持される受け渡しデータは，データを受領するサブシステム側で使用後に削除すること。
- FTP(S)で送信したファイルは，ファイルを送信するサブシステム側が削除すること。
- FTP(S)で受信したファイルは，ファイルを受信するサブシステム側が使用後に削除すること。
- サブシステムから外部システム連携層への通信は同期で連携すること[31]。

[30] 詳細は，『データ統合方針書』の「2.3 データ配置の考え方」を参照のこと。

[31] サブシステムと外部システムとの連携において，内部システムで扱うデータは，基本的にリアルタイム連携による業務単件データが中心であるため，外部システムに対し，業務単件データをまとめて連携する場合，連携タイミングにギャップが発生する。このような連携タイミング等のギャップを吸収するために，連携先の業務要件に応じて同期で連携するか非同期で連携するかを外部システム連携層で定めることになる。アーキテクチャ標準仕様ではサブシステムからESBへの通信についてのみルールとして定める。

(3)　処理方式のイメージ

外部システム連携層をESBと外部システム互換機能で実現する場合の処理方式のイメージを下図に示す。

下図については，外部システムが刷新される際に外部システム連携層を廃止することを想定し，プロセスデータ（図中の「受け渡し情報（引数）」）のみを授受して，処理に必要なデータを共有DBから取得する方式としている。

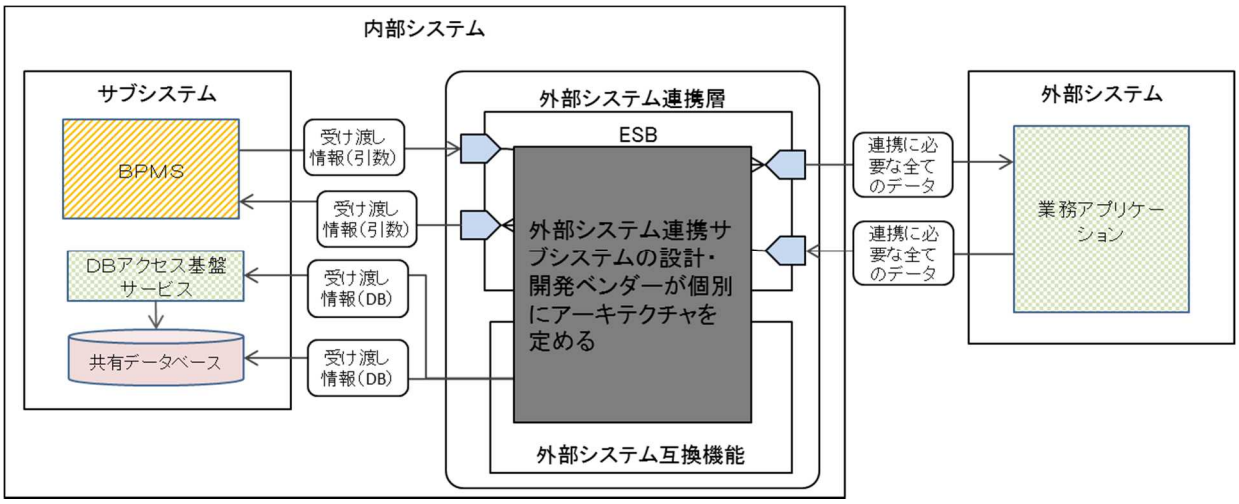


図 3.1-42 外部システム連携層を介した外部システム連携の処理イメージ

(4)　ESBに関する特記事項

前述の「図 3.1-42」に関連し，外部システム連携層にESBを用いる場合の特記事項を，以下に示す。

- ESBによる外部システム又は他のシステム構成要素へのアクセスは以下の場合に限る。
  - 内部システムから受信したデータを連携するために外部システムへアクセスする。外部システム刷新後の連携方式がワークフロー連携になる場合，内部システムからESBへはプロセスデータを引数で受け渡す。
  - 外部システムから受信したデータを連携するためにBPMS，BPMS補完機能，業務アプリケーション（システム），DBアクセス基盤サービスへアクセスする。外部システム刷新後の連携方式がワークフロー連携になる場合，ESBからBPMS，BPMS補完機能，業務アプリケーション（システム）へはプロセスデータを引数で受け渡す。
  - 内部システムと外部システムとの連携方式のギャップをESBで吸収できない場合に限り，ギャップを吸収する処理を行うために，外部システム互換機能へアクセスする。

(5)　外部システム互換機能に関する特記事項

前述の「図 3.1-42」に関連し，外部システム連携層に外部システム互換機能を用いる場合の特記事項を，以下に示す。

- 外部システム互換機能による他システム構成要素へのアクセスは以下の場合に限る。
  - ESBで処理できない連携方式のギャップを吸収する処理で，共有データベースへのデータ取得又は更新のために，DBアクセス基盤サービス又は共有データベースへアクセスする。内部システムから外部システムへ渡すデータにプロセスデータ以外のデータがある場合は，外部システム互換機能がDBアクセス基盤サービス又は共有データベースにアクセスして必要なデータを取得する。また，外部システムから内部システムへ渡すデータにプロセスデータ以外のデータがある場合は，外部システム互換機能がDBアクセス基盤サービス又は共有データベースにアクセスして外部システムから取得したデータを格納する。

なお，ESBで処理できない連携方式のギャップを吸収する際にデータ蓄積が必要な場合等のために，外部システム互換機能の内部にデータベースを持ってもよい。

3.1.2.6.2 データの保護方式

連携処理方式のデータの保護方式は，オンライン処理方式のデータの保護方式と同様とする。
詳細は「3.1.2.2.3 データの保護方式」を参照のこと。

3.1.2.6.3 業務継続方式

連携処理方式の業務継続方式は，オンライン処理方式及びバッチ処理方式の業務継続方式と同様とする。

詳細は「3.1.2.2.5 業務継続方式」，「3.1.2.3.5 業務継続方式」を参照のこと。

なお，BPMSやESB等のプログラムプロダクトについては，導入する製品によって業務継続方式が異なるため，再実行の方法等を個別に設計するものとする。

3.2 多階層構造の層の境界において遵守すべきルール

3.2.1 インタフェースのルールとして定める範囲

インタフェースには内部インタフェースと外部インタフェースがある。サブシステム内インタフェース，サブシステム間インタフェース及びサブシステムと外部システム連携サブシステム間インタフェースを内部インタフェース，外部システムと外部システム連携サブシステム間インタフェースを外部インタフェースと呼ぶ。

インタフェースの種類を図示すると，以下のとおりとなる。

本章節で定めるルールの対象は内部インタフェースである。外部インタフェースに対してルールを設け，制約を課す事は出来ないため，外部インタフェースについては本章節で定めるルールの適用対象外とする。

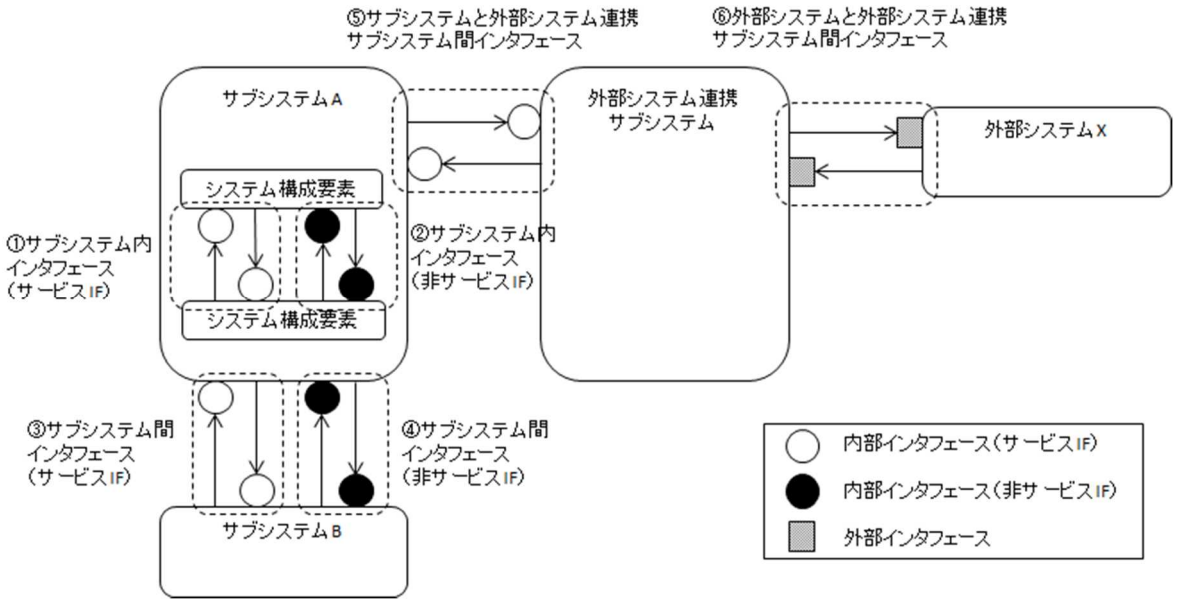


図 3.2-1 インタフェースの種類

また，内部インタフェースには，他のサブシステムに対して公開しているものと非公開のもの，サービス化されるものとサービス化されないものが存在する。

これらを踏まえ，インタフェースの種類とそのインタフェースを提供するシステム構成要素との関係について，以下の表にまとめる。

表 3.2-1 インタフェースの種類とシステム構成要素との関係

項番	インタフェースの種類	サービス／非サービス	インタフェースを提供するシステム構成要素	インタフェースの種類の図例（図 3.2-1）
1	内部インタフェース	サービス	BPMS BPMS補完機能	①，③，⑤のいずれか
2			業務アプリケーション（システム）	①，③，⑤のいずれか
3			DBアクセス基盤サービス	①，③，⑤のいずれか
4			ESB 外部システム互換機能 （いずれも内部システムとのインタフェース）	⑤
5			BRMS	①，③のいずれか
6		非サービス	プレゼンテーションロジック	②
7			業務アプリケーション（バッチ）	②，④[32]のいずれか
8			個別データベース	②
9			共有データベース	②

[32] 業務アプリケーション（バッチ）によるサブシステム間連携処理方式において，FTP(S)を用いたインタフェースが存在する。詳しくは「3.1.2.6.1.3 バッチ処理方式のサブシステム間連携」を参照のこと。

項番	インタフェースの種類	サービス／非サービス	インタフェースを提供するシステム構成要素	インタフェースの種類の図例（図 3.2-1）
10	外部インタフェース	サービス	ESB 外部システム互換機能 （いずれも外部システムとのインタフェース）	⑥
11		非サービス	ESB 外部システム互換機能 （いずれも外部システムとのインタフェース）	⑥

上記表中の項番1～5のサービス化されるインタフェースの考え方については「3.2.1.1.1 サービス化の指針」にて，項番1～9の内部インタフェースのプロトコルに関する適用方針については「3.2.1.1.2 内部インタフェースのプロトコル」にて記載している。

なお，項番10，11の外部インタフェースについては，前述のとおり本章節で定めるルールの適用対象外とする。

3.2.1.1 内部インタフェースのルール

3.2.1.1.1 サービス化の指針

目的:	SOAを導入することにより，環境変化に対して短期間で迅速かつ柔軟にシステムを対応させることを可能とする。また，サービスの提供する機能の再利用を促進し，類似機能の重複開発を防ぐ。更に，インタフェースが変わらない限り，サービスの内部構造の変更による影響がサービスの利用者側に波及することを抑止し，交換可能性を確保する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	「(4)サービス化の対象」に示すシステム構成要素の提供機能は，サービスインタフェースを設けるよう設計すること。
特例1:	「(3)サービス化対象外の指針」に該当するものについては，サービス化の対象外とすることを許容する。
規約2:	サービスインタフェースは，「(5)サービスインタフェースの提供」のとおり提供すること。

(1) サービスの条件

サービスは，条件として下記のようなインタフェースを提供しなくてはならないものとする。

- 共通的に利用されることを想定し，インタフェース仕様が管理されている。
- 予め定められた統一的な方法によって利用側からアクセスすることができる。
- 業界標準のプロトコルを用い，ネットワーク経由でアクセスできる。
- 実装プラットフォームや開発言語の影響を受けずにアクセスできる。

(2) サービス化の指針

サービスとするためにシステム構成要素に一つ以上のサービスインタフェースを設けることを「サービス化」と呼ぶ。

サービス化の指針を以下に示す。

- 他のサブシステムからアクセスされるシステム構成要素をサービス化する。
- サブシステム内で再利用性，交換可能性が求められるシステム構成要素をサービス化する。

(3) サービス化対象外の指針

業務特性や技術的理由により，サービス化しない方がよい場合もある。サービス化の対象外とする指針を以下に示す。

以下の条件に当てはまる場合は，サービス化の対象外とする。

- 「(1)サービスの条件」で示す条件を実現できない場合
- リアルタイム性が求められサービス化により要求性能が満たせない場合
- サービスにアクセスするオーバヘッドが処理時間の多くを占めることにより要求性能が満たせない場合
- サービス化により信頼性要件を満たすための実現コストが許容できない場合

※例えば，複数のサービスをまたがって処理が実行される場合，アクセスするサービス数が多いほど信頼性は低下する。高い信頼性を確保するためにはハードウェアのコスト増につながる可能性がある。

- 複数のサービスにまたがるトランザクション処理や復旧処理が必要となったり，処理のオーバヘッドによって要求性能が満たせない場合
- 複数の分かれた操作にまたがってACID特性が求められるような場合

(4)　サービス化の対象

サービス化の指針を踏まえたサービス化対象を下表に示す。

ただし，「(3)サービス化対象外の指針」に該当する場合は，特例としてサービスとしないことを許容する。

表 3.2-2 サービス化対象

項番	サービス化対象のシステム構成要素	提供機能
1	BPMS	ワークフローの開始，イベント通知，ワークフローの状態取得，ユーザタスクの進行等
2	BPMS補完機能	ワークフローの開始，イベント通知，ワークフローの状態取得，ユーザタスクの進行等
3	業務アプリケーション（システム） ※業務単位のサービスについては，「3.3.3.2 業務アプリケーション（システム）のサービスインタフェースの粒度 (1)サービスインタフェースの粒度のルール」参照 ※サービスインタフェースの利用を許可するシステム構成要素については「3.1.1.3.1.3 (2)」参照。	BPMSのサービスタスクに対応するビジネスロジックの実行。 ※業務単位のサービス
4		BPMSサービスの補完処理に必要なデータ取得及び更新のためのビジネスロジックの実行。
5		個別データの取得及び更新のためのビジネスロジックの実行。
6		共通リソースデータの取得及び更新のための共通的に利用するビジネスロジックの実行。
7	BRMS	ビジネスルールの実行
8	ESB	外部システムとの連携
9	外部システム互換機能	外部システムとの連携
10	DBアクセス基盤 サービス	共有データベースへの参照・更新 ※一つの操作にトランザクションが閉じている操作。

(5)　サービスインタフェースの提供

サービスインタフェースを以下のとおり提供すること。

- セッションを使用しない[33]サービスインタフェースを提供すること。

[33] ここでいうセッションは，ServletのHttpSession等，特定の実装手段に限らず，同じ送信元からの複数のリクエストであることを識別するために，ある定まった期間に行われる一連の通信をひとまとまりのものとして扱う仕組み全般を指す。

3.2.1.1.2 内部インタフェースのプロトコル

目的:	業界標準の通信プロトコルとデータフォーマットを使用することにより，再利用性及び移植性を高め，ベンダロックインの排除を促進する。また，サブシステム間での認識齟齬を防止する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	「表 3.2-3 プロトコル及びデータフォーマット（階層定型化サブシステム）」及び「表 3.2-4 プロトコル及びデータフォーマット（インタフェース定型化サブシステム）」に示すシステム構成要素は，当該表に示すプロトコル及びデータフォーマットを利用するインタフェースを設けること。
特例1:	「表 3.2-3 プロトコル及びデータフォーマット（階層定型化サブシステム）」に示す特例を許容する。

(1) プロトコルの適用指針

内部インタフェースのプロトコルの適用指針を以下に示す。

- 原則，同期呼び出しでステートレスなプロトコルとして最も一般的に利用されているプロトコルであるHTTP(S)を利用する。
  - 大容量ファイルを効率よく転送したい場合や，一括処理のため複数件数のレコードデータを一つのファイルにまとめてやり取りしたい場合等，HTTP(S)では，効率が悪い場合は，ファイル転送プロトコルとして最も一般的に利用されているFTP(S)の利用を許容する。
  - HTTPSやFTPSは，セキュリティ上，データを保護したい場合に使用する。

- 特定の用途で利用される業界標準のプロトコルが存在する場合は，そのプロトコルを利用する。
  例えば，ユーザ情報を取得する機能（認証機能等）は，業界標準であるLDAPを利用する。
  なお，保守性や移植性よりも，効率性，信頼性等の品質特性を優先する場合は，製品が定めるプロトコルをそのまま利用することを許容する。

(2) HTTP(S)のデータフォーマットの適用指針

HTTP(S)のデータフォーマットは，スキーマ定義により電文仕様を厳密に定義できることでサブシステム間の仕様の認識齟齬を防ぐ効果が期待できるXML形式を利用する。

ただし，以下の場合，他のデータフォーマットも許容する。

- アクセス元がブラウザの場合，Webページの表示のためHTML及びCSSを利用する。また，Ajaxを利用して非同期的にデータを授受する場合はXML形式だけでなく，JSON形式を許容する。
- UI層とプレゼンテーション層間の通信では，以下の場合に限りセッションの利用を許容する。
  - 認証・認可に関する情報を保持する場合
  - 一時的に使われる，秘匿性が高い情報をリクエスト間で受け渡す場合
  - 処理の冗長化や性能低下を防ぐ目的で一時的に使われる情報を保持する場合（ユーザの入力値や検索結果の業務処理結果を保持等）
  - 業務的要件，システム的要件を実現するためにセッション保持が必要となる場合（不正な画面遷移を防止する目的で，識別子の保持にセッションを使用する等）

(3) 階層定型化サブシステムが提供するインタフェースのプロトコル及びデータフォーマット

「(1)プロトコルの適用指針」及び「(2)HTTP(S)のデータフォーマットの適用指針」に基づき，階層定型化サブシステムが提供するインタフェースのプロトコル及びデータフォーマットを下表に示す。なお，業務システム(ユーザ)はプレゼンテーションロジックからのメソッド呼び出しとなるためプロトコルを定義する対象外である。

表 3.2-3 プロトコル及びデータフォーマット(階層定型化サブシステム)

項番	インタフェースの提供側システム構成要素	プロトコル	データフォーマット	使用条件
1	プレゼンテーションロジック 使用条件: ブラウザからアクセスする場合	HTTP(S)	HTML	Ajaxを利用して非同期にデータを授受する場合
			CSS	
			XML(※特例)	
			JSON(※特例)	
			HTML	上記以外の場合
			CSS	
2	プレゼンテーションロジック 使用条件: リッチクライアントからアクセスする場合	HTTP(S)	XML	―
3	BPMS BPMS補完機能	BPMS製品が使用するプロトコル	BPMS製品が使用するデータフォーマット	情報活用系サブシステムへ情報を提供する目的で,ETLからアクセスする場合
		ワークフローAPIが定めるプロトコル (※特例)	ワークフローAPIが使用するデータフォーマット(※特例)	BPMS製品が提供するワークフローAPIを使用しアクセスしなければならない場合 (例:ヒューマンタスクのステータスを参照・更新する場合)
		HTTP(S)	XML	上記以外の場合
4	業務アプリケーション(システム)	HTTP(S)	XML	―
5	業務アプリケーション(バッチ)	FTP(S)	テキストデータ (例:CSV)	以下の条件をすべて満たす場合 ● 他サブシステムの業務アプリケーション(バッチ)からアクセスする場合 ● データベース経由では効率が極めて悪い場合 (例:大容量ファイルを効率よく転送したい場合や一括処理のため複数件数のレコードデータを一つのファイルにまとめてやり取りしたい場合)
			特定の用途で利用される業界標準のデータフォーマット (例:JPEG, TIFF, ZIP)	
6	個別データベース	RDBMS製品が使用するプロトコル	RDBMS製品が使用するデータフォーマット	―
7	DBアクセス基盤サービス[34]	HTTP(S)	XML形式	―
			バイナリ形式	
8	共有データベース[35]	RDBMS製品が使用するプロトコル	RDBMS製品が使用するデータフォーマット	―

[34] 詳細は,『DBアクセス基盤 利用者向けガイドライン 5.基盤サービス』を参照のこと。
[35] 業務アプリケーション層の業務アプリケーションからアクセスする場合は,「3.2.1.1.1(3)サービス化対象外の指針」に該当するため,共有データベースへ直接アクセスする場合に採用するプロトコル及びデータフォーマットになる。

(4) インタフェース定型化サブシステムが階層定型化サブシステムに提供するインタフェースのプロトコル及びデータフォーマット

「(1)プロトコルの適用指針」及び「(2)HTTP(S)のデータフォーマットの適用指針」に基づき，インタフェース定型化サブシステムが階層定型化サブシステムに提供するインタフェースのプロトコル及びデータフォーマットを下表に示す。

表 3.2-4 プロトコル及びデータフォーマット(インタフェース定型化サブシステム)

項番	インタフェースの提供側システム構成要素	プロトコル	データフォーマット	使用条件
1	外部システム連携	HTTP(S)	XML	
		FTP(S)	テキストデータ(例:CSV)	以下の条件をすべて満たす場合 ● 他サブシステムの業務アプリケーション(バッチ)からESBにアクセスする場合 ● HTTP(S)では効率が極めて悪い場合(例:大容量ファイルを効率よく転送したい場合や，一括処理のため複数件数のレコードデータを一つのファイルにまとめてやり取りしたい場合)
			特定の用途で利用される業界標準のデータフォーマット(例:JPEG，TIFF，ZIP)	
2	リソース管理	LDAP	LDAPに対応するデータフォーマット	―
3	ビジネスルール管理	HTTP(S)	XML	―

(5)　製品が定めるプロトコルの利用制限事項

製品が定めるプロトコルは,「表 3.2-3 プロトコル及びデータフォーマット(階層定型化サブシステム)」に示すシステム構成要素に使用される場合がある。

製品が定めるプロトコルをそのまま利用する場合，業務アプリケーション内のビジネスロジックが製品の提供する通信ライブラリの独自APIを直接呼び出し，特定の製品に依存してしまうといったことがないようにするため，呼び出し方を制限する(「表 3.1-17 APレイヤ」の項番3)。

3.3 多階層構造の層毎に遵守すべきルール

3.3.1 UI層及びプレゼンテーション層のルール

(1) UI層

UI層の責務及びUI層のシステム構成要素(ブラウザ, リッチクライアント)の責務を下表に示す。

なお,「図 3.1-11」にて示すとおり, 特例としてUI層に配置されるシステム構成要素(業務アプリケーション(ユーザ))が存在する。このシステム構成要素の責務と主な処理内容については, 後述の「表 3.3-11」にて説明を行っている。

表 3.3-1 UI層とシステム構成要素の責務

UI層の責務	システム構成要素の責務	
	ブラウザ	リッチクライアント
ユーザに対して入出力を伴う直接的なインタフェース(GUI)の提供	必要な情報(HTML, CSS, 画像ファイル等)を都度Webサーバから取得・解釈する, ユーザへの画面機能の提供	必要な情報(リッチクライアント固有の形式)を全て事前に配備・使用する, ユーザへの画面機能の提供

UI層の責務を担うために行う主な処理内容を以下に示す。

- システム利用者[36]が入力したデータの内容や操作したイベント[37]の有効性の確認。有効でない場合, 画面表示によるシステム利用者への通知
- システム利用者の入力したデータの内容, 操作したイベントに基づく画面の構成
- システム利用者の入力したデータの内容, 操作したイベントに基づく電文の組み立て
- 自サブシステムのプレゼンテーション層へのリクエスト[38]
- プレゼンテーション層からのレスポンスに基づく画面の構成

(2) プレゼンテーション層

プレゼンテーション層の責務及びプレゼンテーション層のシステム構成要素(プレゼンテーションロジック)の責務を下表に示す。

表 3.3-2 プレゼンテーション層とシステム構成要素の責務

プレゼンテーション層の責務	システム構成要素の責務
	プレゼンテーションロジック
UI層とワークフロー層・業務アプリケーション層との連携	UI層からの要求を下位層に振り分け, その実行結果を編集し, UI層へ返却する機能の提供

プレゼンテーション層の責務を担うために行う主な処理内容を以下に示す。

- UI層からのリクエストの受付
- UI層からのリクエストデータの有効性の確認。有効でない場合, UI層へのエラー情報の返却
- UI層からのリクエストデータの内容に基づくワークフロー層(自又は他サブシステム)又は業務アプリケーション層(自又は他サブシステム)へのリクエスト
- ワークフロー層(自又は他サブシステム)又は業務アプリケーション層(自又は他サブシステム)からのレスポンスデータの編集(UI層へのレスポンス用に編集)
- UI層へのレスポンスデータの返却

[36] 庁内の職員のこと。利用環境は, 一般的なPC用ディスプレイを持ち, マウス及びキーボードにより画面操作を行うことが可能な業務用PCとする。

[37] 「ボタンが押されたこと」や「テキストボックスのフォーカスが外れたこと」等, システム利用者による画面の操作によって発生した状態の変化のこと。

[38] サブシステムをまたがった情報を一画面に表示する場合においても, 自サブシステムのプレゼンテーション層へリクエストする。すなわち, 自サブシステムのプレゼンテーション層が他サブシステムから情報を取得し, 一つの画面に編集した情報を返却する。Webブラウザ上よりクロスドメイン通信によって, 他サブシステムのプレゼンテーション層から情報取得することは行わないものとする。

3.3.1.1 クライアント構成

目的:	ブラウザを積極的に選択させることにより，サブシステム構造の定型化を促進する。また，クライアント端末へのAPインストールが不要になることから，変更時の配布コストを低減する。
スコープ:	階層定型化サブシステム
規約1:	クライアント構成は，「ブラウザを用いた構成」を採用すること。
特例1:	「(2) リッチクライアント構成の選定ルール」に該当する場合に限り，「リッチクライアントを用いた構成」の採用を許容する。

(1) クライアント構成の分類

UI層におけるクライアント構成を下表に示す。

表 3.3-3 クライアント構成

項番	クライアント構成	説明
1	ブラウザを用いた構成	システム構成要素の「ブラウザ」で動作するクライアントAPを用いた構成。 ブラウザからサーバに対しての処理要求の結果として，クライアントAPとなる画面及び処理（HTMLやJavaScript等）がWeb／APサーバから都度返却される。
2	リッチクライアントを用いた構成	システム構成要素の「リッチクライアント」で動作するクライアントAPを用いた構成。 ブラウザを介さず，業務用PCのOS上で動作するクライアントAP。ブラウザで扱うことができないソフトウェアの対応が可能。クライアントAPは，事前に業務用PCにアプリケーションが配備される。

クライアント構成のイメージを下図に示す。

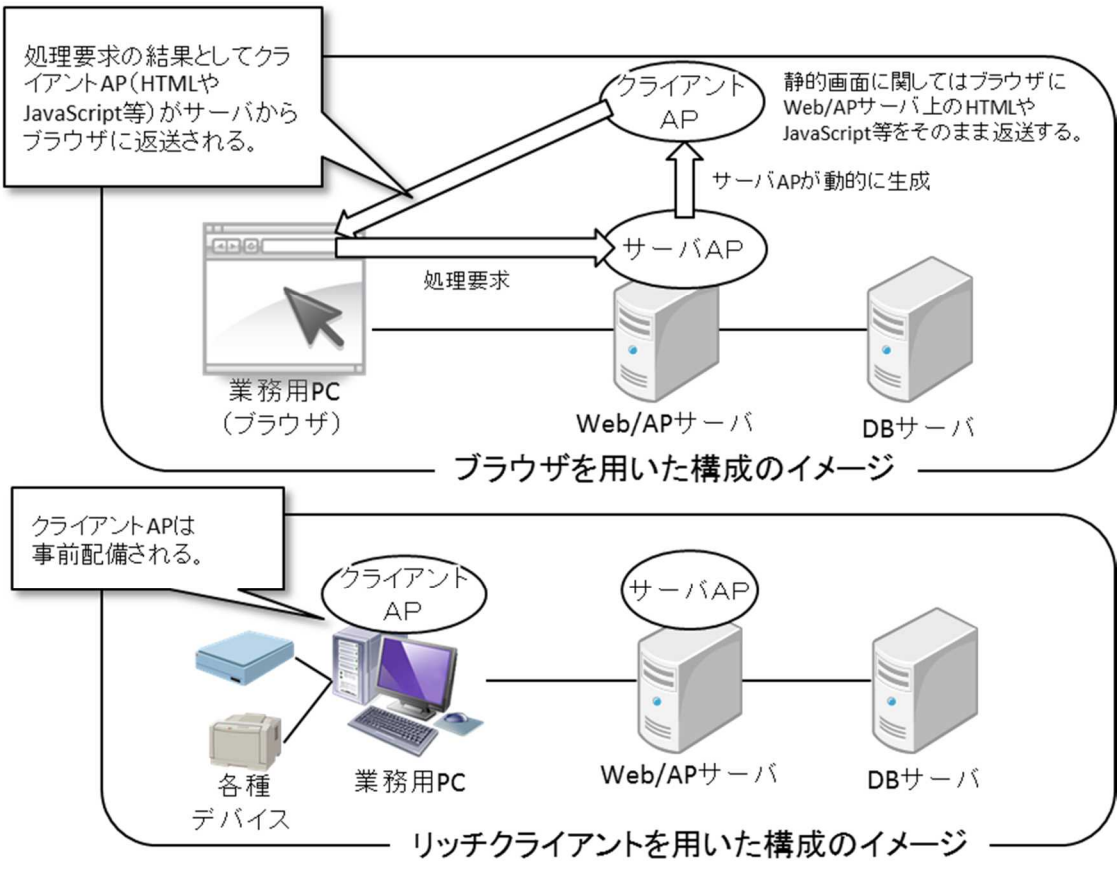


図 3.3-1 クライアント構成のイメージ

保守性を高めるという設計方針を考慮すると，「ブラウザを用いた構成」には「リッチクライアントを用いた構

成」と比べ，以下のメリットがある。

- 幅広い環境で動作し，OSに追加で何らかのプログラムプロダクトをインストールする必要がない。
- アプリケーションは都度サーバから取得し，クライアント側への事前配備を必要としない。

ただし，ブラウザで扱うことのできないソフトウェアを利用する場合等，「リッチクライアントを用いた構成」でないと，実現できない場合もある。

(2) リッチクライアント構成の選定ルール

以下に示す採用条件に該当する場合に限り，「リッチクライアントを用いた構成」を採用すること。

- ブラウザで実現できない高い入力効率が求められる場合
- ブラウザで実現できない表示が求められる場合
- OCRのソフトウェア等，ブラウザで扱うことのできないソフトウェアを利用する場合
- 「ブラウザを用いた構成」では，特定のブラウザに依存した機能を利用しないと実現できない場合

3.3.1.2 画面とリクエスト[39]の作成単位に関するルール

目的:	業務的に理解できる要素(BPMSのアクティビティ)とシステム上対応するアプリケーションの処理単位(画面及びリクエストのまとまり)を対応付けることにより，業務の変更に伴うシステムへの影響範囲の把握を容易にする。
スコープ:	階層定型化サブシステム
指針1:	ユーザタスクに対応する画面とリクエストは，ユーザタスク毎に閉じた単位で作成すること。
特例1:	「(2)画面とリクエストの作成単位に関する特例」に示した業務に限り，複数のユーザタスクに対し同一のリクエストと画面のまとまりを割り当ててもよい。

(1) 指針に関する補足

UI層は，業務の変更による環境変化の影響を受けやすく，改造の頻度が高い。したがって，環境変化に対して影響するリクエストと画面を特定できるように，UI層の業務の切れ目となるBPMSのユーザタスク単位にリクエストと画面のまとまりを割り当て，リクエスト毎に処理(プレゼンテーションロジックと業務アプリケーション(ユーザ))を作成する(下図を参照)。これにより，影響するリクエストと画面を特定しやすくするだけでなく，あるユーザタスクに対応するリクエストと画面に発生した環境変化による影響を，他のユーザタスクのリクエストと画面に波及させないようにする。

なお，上記単位内の画面遷移においては，静的HTMLを全画面において事前に準備しておいたり，クライアント上で動作する簡易プログラム(スクリプト)を作成することにより動的に変更させたり，様々な実装方法が採用し得るが，ルール化することによるメリットが特にないため設計の裁量とする。

また，リッチクライアントについては，その使用を前述(「3.3.1.1 クライアント構成」の「(2) リッチクライアント構成の選定ルール」)のとおり特殊な業務要件に限定することから，リクエストと画面の作成単位が業務要件に強く依存する。したがって，リッチクライアントのリクエストと画面の作成単位についてはルール化せず設計の裁量とする。

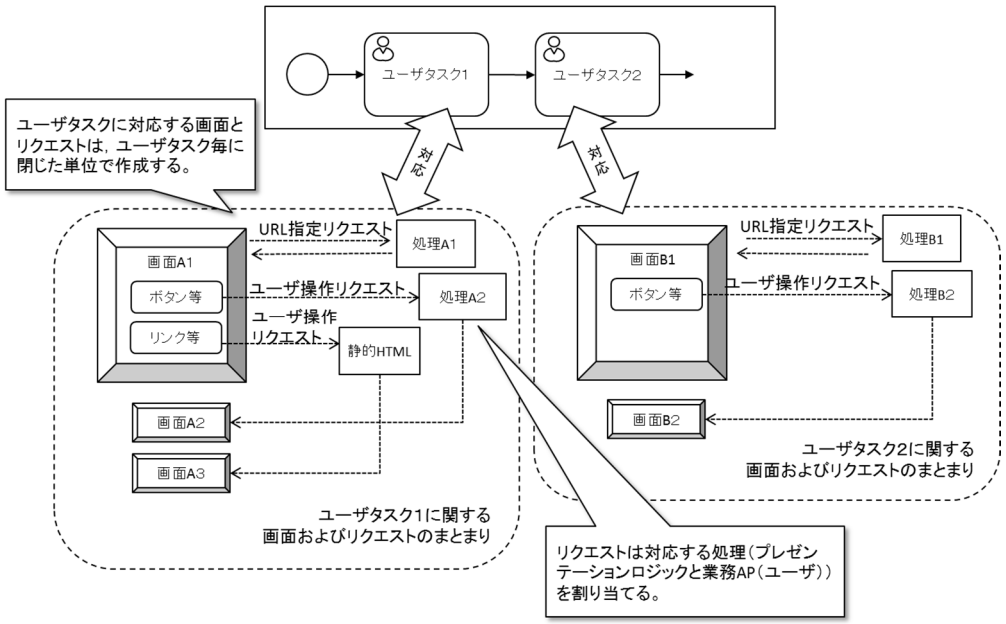


図 3.3-2 ユーザタスクに対するリクエスト及び画面の作成単位

[39] ここでいうリクエストとは，ブラウザからの要求を受けて応答を返すWebサーバのHTTPプロトコルインタフェースによるやり取りのことである。具体的には，プレゼンテーションロジックにおいてルールとして許容されているインタフェースによるやり取り(Ajaxによるアクセス，動的HTMLをロードするためのAPIの呼び出し，単なる静的HTMLのロード等)のことを指す。

(2)　画面とリクエストの作成単位に関する特例

以下の業務処理に限り，複数のユーザタスクに対し同一のリクエストと画面のまとまりを割り当ててもよい。これらの具体例については，『特実審査周辺システム概念設計書』の「6.1.5 サブシステムを横断した画面表示に伴う連携」を参照すること。

- 複数のユーザタスクを同時に進める業務処理

例えば，国際調査と国際予備審査のセット起案という，作成される起案書類をまとめて起案及び決裁する必要があるような業務では，起案及び決裁を一つのUIから実現するのが効率的である。この二つの業務が別のサブシステムに分かれた場合，ユーザタスクも別になることから，指針に従うと一つのUIでの実現ができなくなる。このような，複数のユーザタスクとして分離される業務を同時に扱う必要がある場合においては，特例としてこれらユーザタスクの群単位にリクエストと画面のまとまりを割り当てることを許容する。

- 複数の業務データを横断的に参照する業務処理

例えば，公報発行の状況や運用履歴を照会する際に，法域ごとに画面を立ち上げて照会するよりも，一つの画面を使用して複数の法域のデータをまとめて照会できた方が便利である。サブシステムが法域毎に分割された場合でも，このようなUIの実現を可能とするために，特例として許容する。

(3)　補足

基本的には指針に従い，ユーザタスクとリクエスト及び画面のまとまりとの対応関係がとれるようにすることが望ましい。したがって，特例の適用にあたっては，サブシステム分割やアクティビティの粒度が適切かどうかも合わせて検討すること。

3.3.1.3 画面の定義に関するルール

目的:	業務的に理解できる要素(BPMSのアクティビティ)とシステム上対応する画面の単位を対応付けることにより，業務の変更に伴うシステムへの影響範囲の把握を容易にする。
スコープ:	階層定型化サブシステム
規約1:	画面の単位は，「(1)画面の単位」に従って設計すること。
規約2:	画面間の遷移は，「(2)画面間の遷移」に従って設計すること。

(1) 画面の単位

業務的な画面の単位を決定する際のルールを以下に示す。

なお，画面を制御するためのビジネスロジック等のコンポーネントを共通化することを制限するものではない。

- アクティビティ(ユーザタスク)をまたがらないよう画面を分けること。

ToBeアーキテクチャでは，BPMSによってアクティビティの実行順序を制御する。ビジネスプロセスの変更によりアクティビティの順序に変更があった場合でも画面構成等への影響を受けにくくするため，画面がアクティビティ間の順序性を持たないよう，アクティビティ毎に画面を分け，アクティビティをまたがる単位で画面を作成しないこと。

例: 起案，決裁

ただし，ダイアログのようにアクティビティが異なっても共通的に利用される画面で，画面構成等の将来的な変更内容がアクティビティ毎に異ならないことが想定される場合は，同じ画面として定義してもよい。

- 機能をまたがらないよう画面を分けること。

一つの画面に複数の機能が組み込まれると，画面の目的が曖昧になり操作性が低下するため，参照や更新等の機能毎に画面を分け，機能をまたがる単位で画面を作成しないこと。

例: 案件照会，書類照会，起案

(2)　画面間の遷移

画面間の遷移に関するルールを以下に示す。

- 他のアクティビティに関する画面群に直接遷移しないこと。

アクティビティの実行順序はBPMSが制御するため，画面側はその順序が変更になっても影響を受けないようにするため，他のアクティビティに関する画面群に直接遷移せず，トップ画面から遷移するようにする[40]。

画面間の遷移の例を下図に示す。

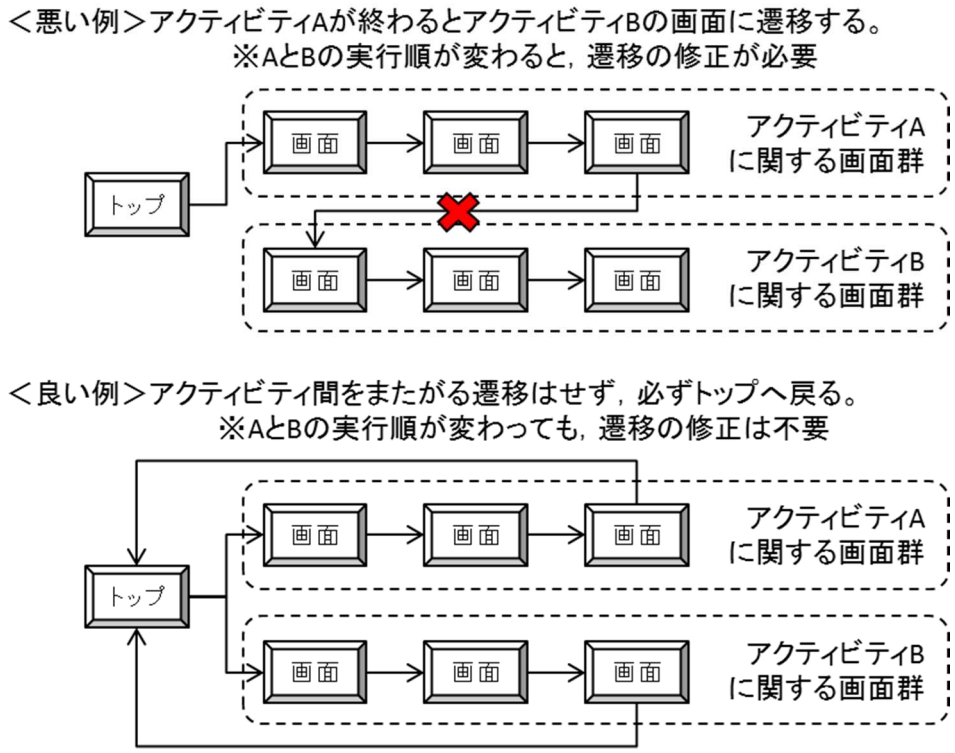


図 3.3-3 画面間の遷移の例

あるアクティビティを完了させ次のアクティビティに進めることを例とすると，以下のような制御となる。

① 画面群における最初の画面で，特定のステータスである対象の一覧を表示する。
② システム利用者がステータスを先に進める対象を一覧から選択すると，一覧画面から必要な情報の入力等を行う画面に遷移する。
③ 画面群の最後の画面のボタンアクション等により情報の入力を完了させると，画面群が閉じてステータスが次に進む。

なお，「アクティビティに関する画面群内」の画面遷移は，別画面へ遷移する方式と画面内の一部分を動的に更新する方式がある。

[40] ここの画面間の遷移は，画面上のリンクやボタンで行うことを前提としている。

3.3.1.4 ブラウザからの要求を受けるアプリケーション[41]の粒度[42]

目的:	複数のサブシステムをまたがらないようにブラウザからの要求を受けるアプリケーションを構成することにより，サブシステム構造の定型化を図る。また，変更による影響範囲を個々のサブシステム内に局所化する。これらの事により，アプリケーションの配置（どこに置かれているか）の把握を容易にし，システムの運用や保守をアプリケーションごとに独立して管理することにより運用・保守の柔軟性を向上する。
スコープ:	階層定型化サブシステム
指針1:	ブラウザからの要求を受けるアプリケーション（プレゼンテーションロジック及び業務アプリケーション（ユーザ）のまとまり）の粒度は，「(1)ブラウザからの要求を受けるアプリケーションの粒度のルール」に従って設計すること。

ブラウザからの要求を受けるアプリケーションはプレゼンテーションロジック及び業務アプリケーション（ユーザ）から構成される。これらの粒度のルールを以下に示す。

ブラウザからの要求を受けるアプリケーションは，リクエストに対する処理（プレゼンテーションロジックと業務アプリケーション（ユーザ））と画面のまとまりを，更に業務のまとまりで束ねたものである。画面とリクエストの作成単位については，「3.3.1.2 画面とリクエストの作成単位に関するルール」を参照のこと。

(1) ブラウザからの要求を受けるアプリケーションの粒度のルール

ブラウザからの要求を受けるアプリケーションは，以下の粒度で構成すること。

- 複数のサブシステムをまたがらないように構成すること。
- サブシステム内においても，「A.分割の観点」に示す観点でアプリケーションを分割すること。

ブラウザからの要求を受けるアプリケーションの粒度のイメージを，以下に示す。

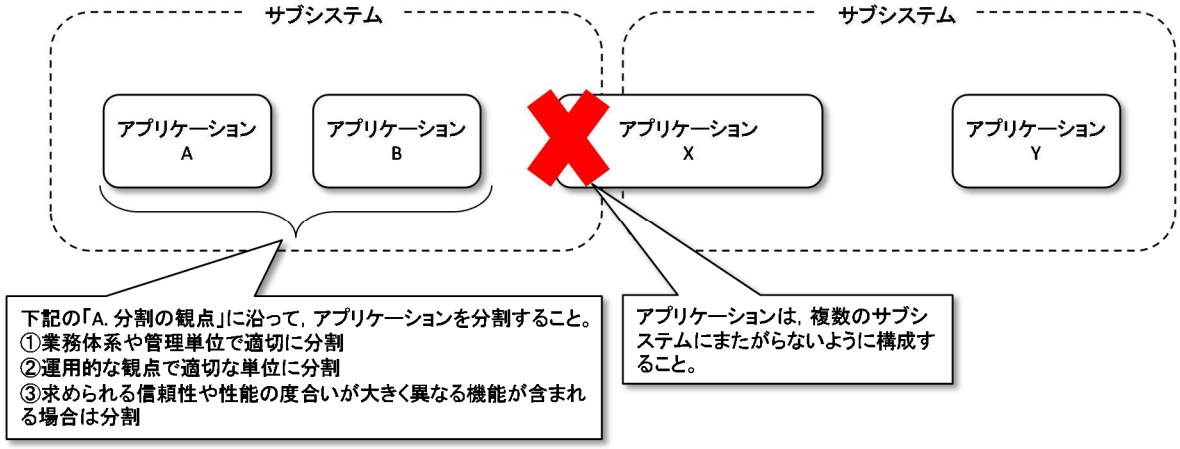


図 3.3-4 ブラウザからの要求を受けるアプリケーションの粒度のイメージ

A. 分割の観点

① 業務体系や管理単位で適切に分割すること。

大きなアプリケーションの管理は人間にとって煩雑なものになる。一方，細かいアプリケーションに分割するとアプリケーションの数が多くなり，同様に管理が煩雑になる。業務の責務の切れ目や管理単位でアプリケーションを分割し，アプリケーションの配置が人間にとって，直感的に把握しやすく，管理しやすい大きさにする。

---

[41] 本章節「3.3.1.4 ブラウザからの要求を受けるアプリケーションの粒度」内に記す「アプリケーション」とは，「アプリケーション（プレゼンテーションロジック及び業務アプリケーション（ユーザ）のまとまり）」のことを指す。

[42] BPMSからの要求を受けるアプリケーション（業務アプリケーション（システム）のまとまり）の粒度については，「3.3.3.1 サービスの粒度」を参照のこと。

② バージョンアップによるアプリケーションの起動・停止等，運用的な観点で適切な単位で分割すること。

サービス提供時間帯が異なる等，運用上独立しているアプリケーションは資材及びデプロイ先を分け独立させることで，システム運用における柔軟性が向上する場合，分割を行う。

③ 求められる信頼性や性能の度合いが大きく異なる機能が含まれる場合は，アプリケーションを分割すること。

アプリケーションが要求するシステムリソース（CPU，メモリ量，DBコネクションプール数等）を別途与える必要があるアプリケーション群に対しては，アプリケーションを分割しておくこと。物理構成においても，アプリケーションに合わせて分割することで，独立して運用が可能となる。

(2) アプリケーションの粒度のルールを考慮したアプリケーション分割例

サブシステム内でアプリケーションを分割する例としては，実体審査サブシステムにおける実体審査と検索外注業務がある。両者は，業務の責務が分かれており，開発単位や保守・運用における扱いも分けられることから，アプリケーションは分割することが可能である。アプリケーションを分割することで，実体審査と検索外注業務のアプリケーション起動・停止等の運用やバージョンアップ等の保守を独立して管理することができるといったメリットがある。

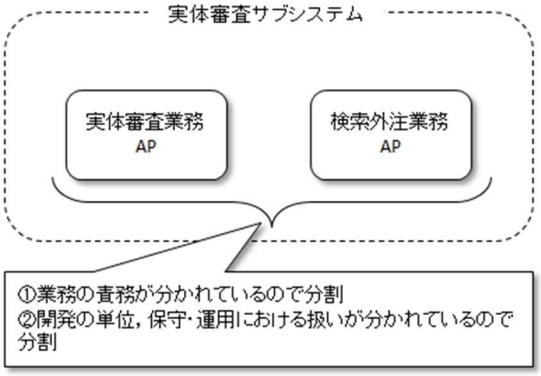


図 3.3-5 アプリケーションの粒度のルールを考慮したアプリケーション分割例イメージ

3.3.2 ワークフロー層のルール

ワークフロー層の責務及びワークフロー層のシステム構成要素（BPMS，BPMS補完機能）の責務を下表に示す。

表 3.3-4 ワークフロー層とシステム構成要素の責務

ワークフロー層の責務	システム構成要素の責務	
	BPMS	BPMS補完機能
業務アプリケーションの連携をビジネスプロセスとして管理及び可視化	● 業務アプリケーションを構成するサービスの実行順序及び実行条件の制御 ● ビジネスプロセスの進行状況の管理	BPMSにおけるワークフロー制御の補完

ワークフロー層の責務を担うために行う主な処理内容を下表に示す。

表 3.3-5 ワークフロー層の主な処理内容

項番	ワークフロー層の主な処理内容	システム構成要素	
		BPMS	BPMS補完機能
1	サービスインタフェースの提供	○	○
2	ワークフローAPI[43]の提供	○	－
3	リクエスト[44]の受付	○	○
4	開始するビジネスプロセス又は連携するビジネスプロセスインスタンスの特定	○	○
5	イベント通知の要否判断の制御	－	○
6	ビジネスプロセスの待機箇所の特定	－	○
7	ビジネスプロセスインスタンスの生成	○	－
8	ビジネスプロセスの開始	○	－
9	ビジネスプロセスインスタンス又はアクティビティの検索[45]	○	－
10	ビジネスプロセスを構成する要素へのイベント通知[46]	○	－
11	ワークフローの定義に基づくプロセストークンを動かす判断等のフロー制御	○	－
12	ワークフローの定義に基づくイベント処理	○	－
13	ワークフローの定義に基づくアクティビティの実行	○	－
14	アクティビティに対応する業務アプリケーション（システム）の実行	○	－
15	ビジネスプロセスの進行状況に関する情報の提供	○	－
16	アクティビティの実行履歴に関する情報の提供	○	－
17	ビジネスプロセス連携の際のプロセスデータの変換	－	○

○：当該処理を担う，－：対象外

[43] ワークフローAPIは呼び出し元の層に配置する。ワークフローAPIとはBPMSのライブラリのことを指し，表中の項番1のサービスインタフェースとは別のものである。BPMSが保持するプロセスデータを検索する，等に用いられる。

[44] ワークフローに対する処理要求のこと。具体的には，ビジネスプロセスの開始やBPMSが保持するプロセスデータの検索，等。

[45] ビジネスプロセスインスタンスの検索とは，キー情報（具体的な内容は「3.3.2.1.3 プロセスデータとして保持する情報」を参照のこと）により対象のプロセスインスタンスを特定するための操作のこと。アクティビティの検索とは，ビジネスプロセスのステータス（ビジネスプロセスがどこのアクティビティにいるか）を特定するための操作のこと。

[46] メッセージイベントに対して待機解除通知を行う，等の操作のこと。

BPMSで行うべき範囲とアプリケーションで行うべき範囲の基本的な考え方として，前者は業務がどこまで進んだかといった進行状況の管理及びプロセストークンを動かす判断（待機判定）を責務とし，後者は進行状況の管理又はプロセストークンを動かす判断（待機判定）は行わず，BPMSからの要求に対する業務処理の実行を責務とする。

業務処理の実行自体は業務アプリケーションの責務だが，実行の契機を与えるのはBPMSである。なお，BPMSからの要求に対する業務処理の実行には，BPMSが待機判定を行うための待機要因[47]に基づいた待機要／不要の導出も含まれる。

[47] ビジネスプロセスがなぜ待機しているのかの要因を示すもの。申請書類の受理等の他のビジネスプロセスのイベントによって，待機要因の内容が変更されたり，新たな待機要因が増えたりする等，刻々と変化する性質を持つため，業務アプリケーションによる導出が必要となる。

3.3.2.1 ビジネスプロセスのルール

3.3.2.1.1 ビジネスプロセス管理として利用する技術

目的:	サービス同士の直接的な連携を排除し，サービス間の関係を疎結合に保つことにより，変更による影響の波及を抑止する。また，業界標準のWebサービスの標準規格にも対応しているBPMSを使用することにより，ベンダロックイン排除する。
スコープ:	階層定型化サブシステム
規約1:	ビジネスプロセスの実行はBPMSで行うこと。

ビジネスプロセスの実行順序や実行条件の制御を一元管理する仕組みとして，BPMSを利用する。[48]

[48] 業務間の順序関係がない単体で完結する業務はビジネスプロセスとして管理する必要はなく，BPMSの利用を強制するものではない。

3.3.2.1.2 BPMSの適用範囲に関する設計指針

目的:	業務的な順序関係がプロセス図に表現されるようにBPMS側で行うべき範囲とアプリケーションで行うべき範囲を定めることにより，BPMSにおけるビジネスプロセスの可視性の低下を防ぐ。
スコープ:	階層定型化サブシステム
指針1:	BPMSの可視性が確保できないケースでのアプリケーションの責務及び責務を担うシステム構成要素は，「表 3.3-7 BPMSで可視性が確保できないケースでのアプリケーションの責務」のとおりとすること。
指針2:	待機制御におけるBPMSとアプリケーションの責務は，「表 3.3-8 待機制御におけるBPMSとアプリケーションの責務の切り分け」に従うこと。
指針3:	各BPMSの呼び出し元における事前の業務チェックを行う箇所は「表 3.3-9 BPMSの呼び出し元と事前チェック箇所」に従うこと。

「3.3.2.1.1」に示したとおり，ビジネスプロセスの制御はBPMSが行う。しかしながら，BPMSのみでビジネスプロセスの制御をするとビジネスプロセスの記載が複雑になり可視性が確保できないケースが存在する。BPMSのみでは可視性が確保できないケースを下表に示す。

表 3.3-6 BPMSのみでは可視性が確保できないケース

項番	ケース		業務例
1	(1) 他のビジネスプロセスとの密接な順序関係を持つビジネスプロセス	A. 予見できないタイミングで発生する異なるビジネスプロセスインスタンスへの割り込み	自発補正
2		B. 再帰的な構造を持ったビジネスプロセスインスタンス間の実行順序制御	補正の補正
3	(2) アクティビティの割り当てが組織構造と密接な関係を持つビジネスプロセス	A. アクティビティに対応するヒューマンタスクの動的な絞込み	実体審査 (予約，起案，決裁)
4		B. 担当者の変更運用	人事異動時の差戻し
5	(3) BPMNの記述上想定外のイベントが発生するケース	A. ビジネスプロセスが開始する前に，中間イベントが通知されるケース	願書を未受理で，出願審査請求書を受理した場合
6		B. イベントゲートウェイを利用し，一方のイベントが通知されフローが進行してしまった後に，他方のイベントが通知されるケース	拒理応答の応答期間経過後，意見書を受理した場合

保守性の向上のためにBPMSにおけるビジネスプロセスの可視性を確保しなければならないという目的(言い換えると，業務の流れをビジネスプロセスで把握できるようにする目的)を踏まえ，「表 3.3-6 BPMSのみでは可視性が確保できないケース」においては，BPMSとアプリケーションを組み合わせてビジネスプロセスを制御する。

この考え方に従い，BPMSで可視性が確保できない場合におけるアプリケーションの責務及び責務を担うシステム構成要素を下表に示す。

表 3.3-7 BPMSで可視性が確保できないケースでのアプリケーションの責務

アプリケーションの責務	責務を担うシステム構成要素	左記の責務に基づいてAPを利用するケースの例	
ビジネスプロセスを待機させるかの判断	業務アプリケーション（システム）	(1)	A.
			B.
プロセスデータやビジネスプロセスの状態（ビジネスプロセスのステータスやヒューマンタスクの状態）の個別データベースへの格納・更新	業務アプリケーション（システム），業務アプリケーション（バッチ）	(2)	A.
			B.
中間イベントに対する通知の可否の判断又はイベントゲートウェイ利用時の通知の可否の判断	BPMS補完機能	(3)	A.
			B.

以下に「表 3.3-6」に示されたケースでアプリケーションを利用した場合のアプリケーションの責務について具体例を挙げて説明する。

(1)　他のビジネスプロセスとの密接な順序関係を持つビジネスプロセス

A.　予見できないタイミングで発生する異なるビジネスプロセスインスタンスへの割り込み

特許庁業務における「自発補正」のように，あるビジネスプロセスインスタンスで発生したイベントにより，既に進行している別のビジネスプロセスインスタンスの処理に割り込み，中断（戻し）したり，待機したりするといった異なるビジネスプロセスインスタンス間での実行順序が必要となるケースがある。このような場合，以下のようにアプリケーションを利用して設計することが考えられる。

- 全てをBPMNで記載せず，ビジネスプロセスインスタンス間の実行順序制御が必要となる適切なポイントで業務アプリケーション（システム）を呼び出し，データベースの最新状態から自ビジネスプロセスを待機させるか判断させる。

これは，ビジネスプロセスとしてのアクティビティの順序関係以外に，粒度が細かいシステムの実装上必要な順序制御のフローが混在することで，ビジネスプロセスの可視性・保守性が低下することを防止するためである。

待機設定，待機判定，待機解除の一連の流れにおける，BPMSと業務アプリケーション（システム）の責務の切り分けについて，以下の表に示す。

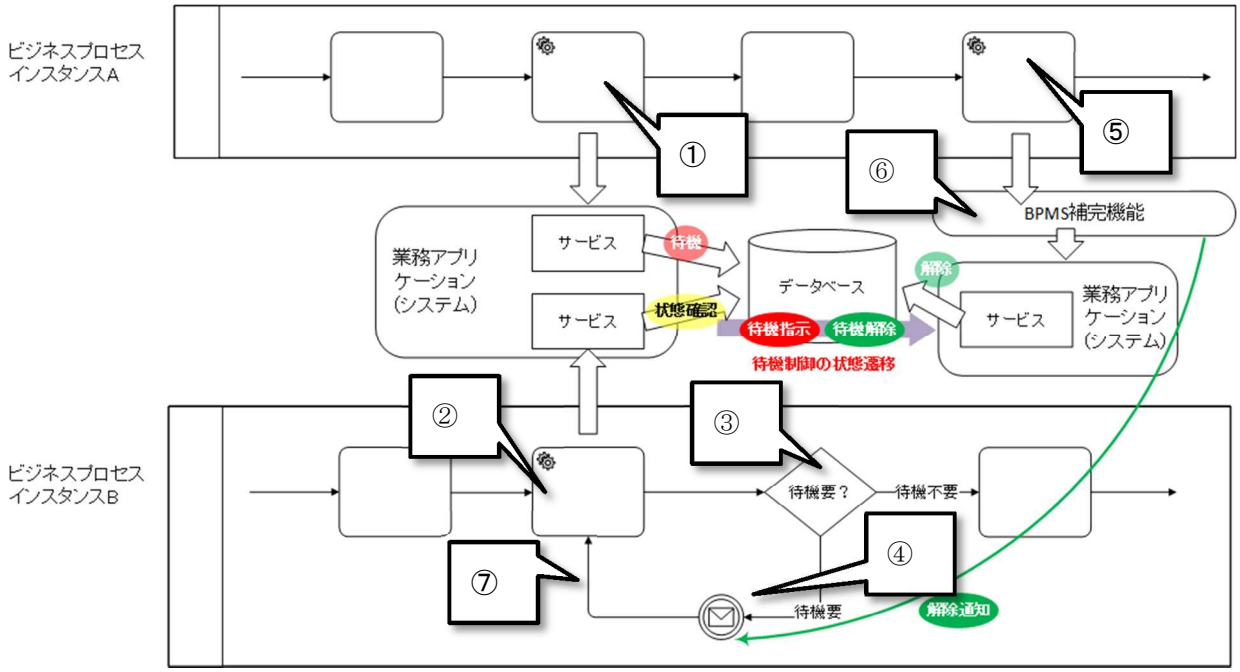
表 3.3-8 待機制御におけるBPMSとアプリケーションの責務の切り分け

項番	場面	BPMSの責務	業務アプリケーション（システム）の責務
1	待機設定	● 待機要求を発生させるビジネスプロセスが，待機制御情報設定サービス（データベースに格納された待機制御情報の更新を行うサービス）の呼び出しを行うこと。	● 待機要否の判断に必要な情報を取得後，待機設定を表すデータは，待機要求を受けるサブシステムの個別DBへ格納すること。
2	待機判定	● 待機要求を受けるビジネスプロセスは，待機設定のデータを取得するサービスの呼び出しを行うこと。 ● 待機要求を受けるビジネスプロセスは，排他ゲートウェイにて待機要否の判定を行い，待機要と判断した場合はキャッチ中間メッセージイベント（あるいはキャッチ中間タイマーイベント）を使い，ビジネスプロセスを待機させること。	● 待機要否の条件判定に用いる待機設定を表すデータの最新状態を取得すること。 ● ユーザタスクにおいて，待機設定が「あり」の場合はその旨を画面に表示してユーザに示すこと。
3	待機解除	● 待機要求を発生させるビジネスプロセスが，待機制御情報設定サービスの呼び出しを行うこと。 ● 待機要求を発生させるビジネスプロセスから呼び出されたBPMS補完機能は，待機要求を受けるビジネスプロセスを特定後，当該ビジネスプロセスのメッセージ待ちに対して待機解除通知をスローすること。 ● 待機要求を受けるビジネスプロセスにおいて，待機解除後は再び待機判定を行うこと。	● 待機解除を表すデータは，待機要求を受けるサブシステムの個別DBへ格納すること。

- 判定条件がビジネスルール化の特性に適すると判断できる場合は，ビジネスルールとして業務アプリケーションから切り離す仕組みを利用する。

ビジネスルールを業務アプリケーションから切り離す仕組みについては，「3.3.6.1.1 ビジネスルールをアプリケーションから切り離すための仕組みとして利用する技術」を参照のこと。

例として，待機制御の処理イメージを下図に示す。



① ビジネスプロセスインスタンス A（待機要求を発生させるプロセス）が業務アプリケーション（システム）のサービスを介し，データベース上の当該案件情報を更新する（待機指示）。
② ビジネスプロセスインスタンス B（待機要求を受けるプロセス）が業務アプリケーション（システム）のサービスを介し，データベースから当該案件の最新情報から待ち要因があるかを取得し，プロセスデータに反映する。（待ち要因は，他のビジネスプロセスのサービスや業務画面からデータベース上の案件情報の更新が行われことにより更新される）。
待ち要因の判定条件がビジネスルール化の特性に適すると判断できる場合は，ビジネスルールとしてアプリケーションから切り離す仕組みを利用すること。
③ ②の結果から排他ゲートウェイで分岐先を判定する。
④ 待機要の場合，キャッチ中間メッセージイベントを使い，通知が来るまで（あるいはキャッチ中間タイマーイベントで一定時間）ビジネスプロセスを待機させる。
⑤ ビジネスプロセスインスタンス A が業務アプリケーション（システム）のサービスを介し，データベース上の当該案件情報を更新する（待機解除）。
⑥ BPMS 補完機能が，ビジネスプロセスインスタンス B のワークフローのメッセージ待ちに対して，待機解除通知をスローする。通知を受け，ビジネスプロセスインスタンス B のワークフローが再開する。
⑦ ②へ戻る。

図 3.3-6 異なるビジネスプロセスインスタンスへの割り込みによる待機制御

なお，排他ゲートウェイと中間イベントを使った待機状態の記載は，ビジネスプロセス上の至るところに発生する。通常シナリオのビジネスプロセスの可読性低下を防止するため，BPMNの記載ルールを別に定めている。詳細は，『別冊2 BPMN記載ルール編』の「1.5 (2) パターン事例2 他ビジネスプロセス起因の待機処理」を参照のこと。

中断制御の場合は，待機制御とほぼ同様の実現方式となる。「図 3.3-6 異なるビジネスプロセスインスタンスへの割り込みによる待機制御」のプロセスインタスタンスBが下図のようになる。

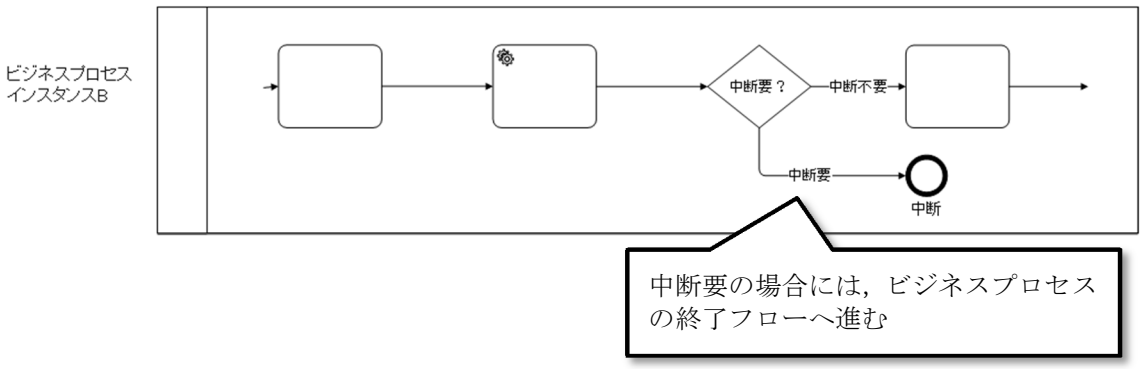


図 3.3-7 異なるビジネスプロセスインスタンスへの割り込みによる中断制御

B. 再帰的な構造を持ったビジネスプロセスインスタンス間の実行順序制御

特許庁業務における「補正の補正」のように，ビジネスプロセスインスタンスが再帰的な構造を持つケースがある。

このような場合，以下のようにアプリケーションを利用して設計することが考えられる。

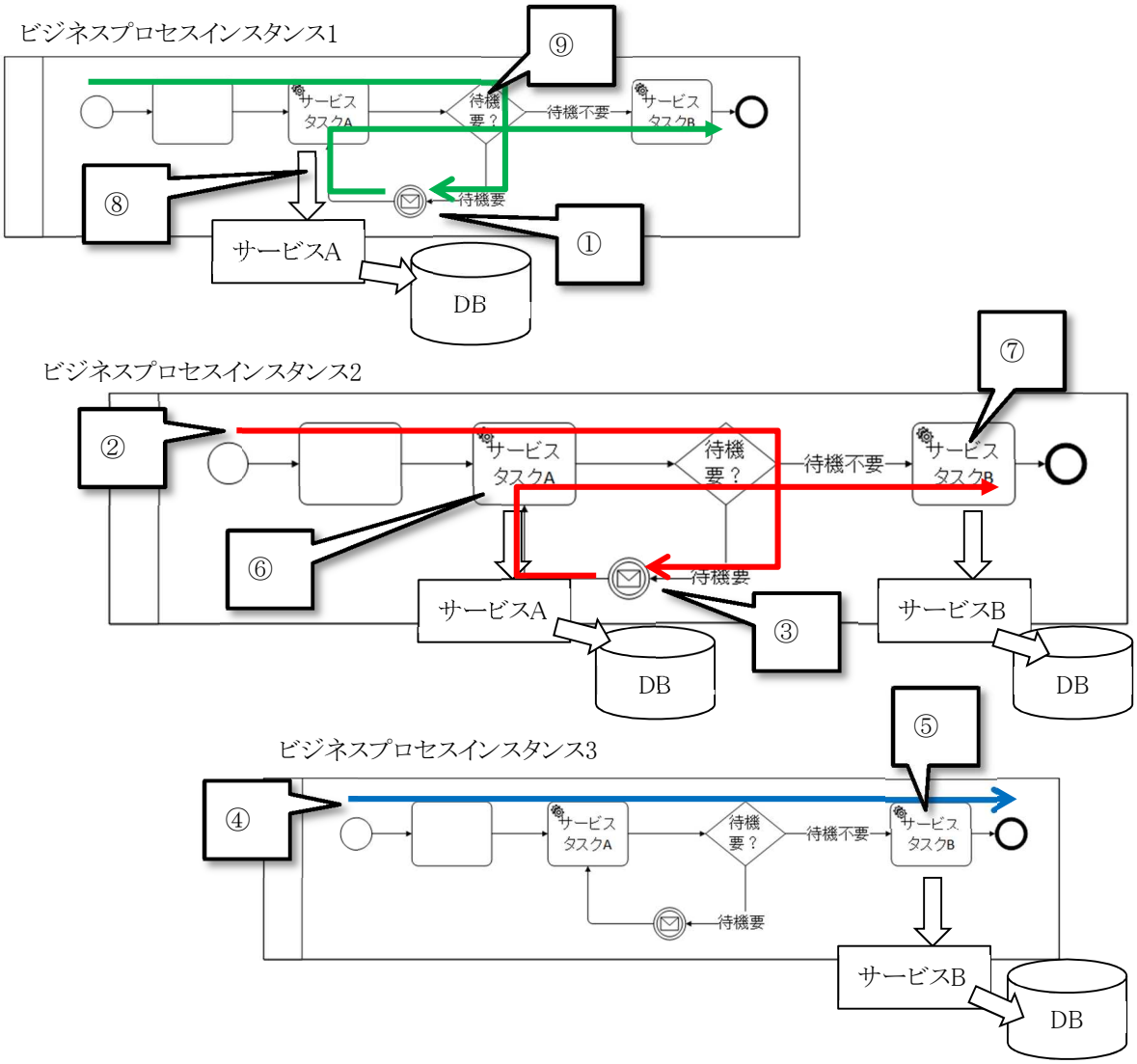
- 全てをBPMNで記載せず，ビジネスプロセスインスタンス間の実行順序制御が必要となる適切なポイントで業務アプリケーション（システム）を呼び出し，データベースの最新状態から自ビジネスプロセスを待機させるか判断させる。

これは，ビジネスプロセスとしてのアクティビティの順序関係以外に，粒度が細かいシステムの実装上必要な順序制御のフローが混在することで，ビジネスプロセスの可視性・保守性が低下することを防止するためである。

- 判定条件がビジネスルール化の特性に適すると判断できる場合は，ビジネスルールとして業務アプリケーションから切り離す仕組みを利用する。

ビジネスルールを業務アプリケーションから切り離す仕組みについては，「3.3.6.1.1 ビジネスルールをアプリケーションから切り離すための仕組みとして利用する技術」を参照のこと。

例として，再帰的な構造を持ったビジネスプロセスインスタンス間の実行順序制御を上記考慮点の一点目を踏まえて処理イメージ図にしたものを以下に示す。



① サービスタスクＡから呼び出したサービスＡの処理結果より判定した結果，待機するフローへ進み，中間イベント（中間タイマーメッセージイベント，キャッチ中間タイマーイベント等）で待機。
② 業務的なイベント発生を契機として新たなプロセスインスタンスが開始。
③ サービスタスクＡから呼び出したサービスＡの処理結果より判定した結果，待機するフローへ進み，中間イベント（中間タイマーメッセージイベント，キャッチ中間タイマーイベント等）で待機。
④ 業務的なイベント発生を契機として新たなプロセスインスタンスが開始。
⑤ サービスタスクＡから呼び出したサービスＡの結果，待機するフローへ進まず，サービスタスクＢのサービスＢを実行し，データベースを更新。
⑥ 待機状態から抜けた後，再度サービスタスクＡからサービスＡを実行し，データベースの最新情報を参照した結果，待機要因が解除されていると判定。
⑦ 再度サービスタスクＡから呼び出したサービスＡの結果，待機するフローへ進まず，サービスタスクＢからサービスＢを実行し，データベース上の案件の情報を更新。
⑧ 待機状態から抜けた後，再度サービスタスクＡのサービスＡを実行し，データベースの最新情報を参照した結果，待機要因が解除されていると判定。
⑨ 再度サービスタスクＡから呼び出したサービスＡの結果，待機するフローへ進まず，次のアクティビティへ進む。

図 3.3-8 再帰的な構造を持ったビジネスプロセスインスタンス間の実行順序制御

なお，再帰的な構造をビジネスプロセスで表す場合，詳細なフローを隠蔽することにより可読性低下を防止する。詳細は，『別冊2 BPMN記載ルール編』の「1.4 ビジネスプロセス記載ルール」を参照のこと。

(2)　アクティビティの割り当てが組織構造と密接な関係を持つビジネスプロセス

A.　アクティビティに対応するヒューマンタスクの動的な絞込み

BPMSでは，BPMNのユーザタスクによって，システム利用者が業務システムの画面を対話的に利用して実行する業務（ヒューマンタスク）を記載できる。ただし，ユーザへのヒューマンタスク割り当て方法は，BPMN仕様だけでは明確に定義されておらず，現状は，製品の実装に依存するところが多い。

BPMS製品の一般的な機能としては，ユーザタスク又はユーザタスクが配置されたBPMNのレーンに対するプロパティ設定で，LDAP等で管理されているユーザやグループをマッピングするといった静的な設定を実施することができる。これにより，製品がサポートする画面機能又は独自に開発した画面APからワークフローAPIを使ってBPMSを呼び出すことで，起動中のビジネスプロセスインスタンスのうち，対象のユーザ又はユーザが属するグループが実行すべき特定のユーザタスク上で待機しているものを一覧取得することができるようになっている。

しかし，特許庁業務には，ユーザタスクやレーンに設定した静的なロール情報による検索だけでなく，動的に変化する業務的な情報（プロセスデータ）による検索が必要な場合がある。

特許庁業務におけるアクティビティに対応するヒューマンタスクの動的な絞込みの例として，実体審査における「予約」，「起案」，「決裁」がある。それぞれ案件を選択するために「予約対象の出願（案件）」，「起案対象の出願」，「決裁対象の出願」を一覧取得する。例えば，「予約対象の出願（案件）」の一覧取得の場合，出願が各審査室（又は審査グループ）に所属する審査官に振分けられるため，審査官全員でなく，振分け先の審査室（又は審査グループ）に所属する審査官のみに「予約対象の出願」の一覧を画面表示する必要がある。

このような検索条件が動的に変化する場合，以下のいずれかの方式により検索条件に合致するアクティビティを絞り込むことができる。[49]

(A)　方式の概要

方式1)

ワークフローAPIを使ってBPMSが保持する業務データ（プロセスデータ）を検索する。

例：「予約対象の出願（案件）」の一覧取得の場合，一覧画面を利用するユーザが所属している審査室（又は審査グループ）を検索条件にして，ワークフローAPIでプロセスデータに格納されている振分け先の審査室（又は審査グループ）を検索する。

方式2)

ワークフローAPIを使ってBPMSが保持する業務データ（プロセスデータ）を検索した結果に対して，更にプレゼンテーションロジックで対象を絞り込む。

例：「予約対象の出願（案件）」で特定の分類情報を持つ出願（案件）を一覧取得する場合，一覧画面を利用するユーザが所属している審査室（又は審査グループ）を検索条件にして，ワークフローAPIでプロセスデータに格納されている振分け先の審査室（又は審査グループ）を検索する。検索した結果に対して，更に，プレゼンテーションロジックで分類情報に合致した出願（案件）を絞り込む。

方式3)

ヒューマンタスクのステータスを個別データベースで二重管理し，個別データベースのみを検索して対象を絞り込む。

例：「予約対象の出願（案件）」の一覧取得の場合，予め振分け業務の際，出願（案件）に対して「予約待ち」というステータスを個別データベースに格納しておき，一覧取得の際は，出願（案件）の「予約待ち」ステータスとユーザが所属する審査室（又は審査グループ）等を検索条件にして個別データベースを検索する。

[49] BPMS製品の機能でアクティビティの絞り込みが可能ならば，BPMS製品の機能を使用してアクティビティの絞り込みを行ってもよい。

(B) 各方式の留意点

以下の方式の留意点を考慮して設計する必要がある。

- ビジネスプロセスの保守性，異常時のリカバリの観点では，方式1を採用することが望ましい。
  ただし，方式1を採用する場合は，以下の点に留意する必要がある。

  - 製品によっては，ワークフローAPIの返却データ形式がXML等のファイルの場合があり，その構文解析処理に多くの時間がかかる場合がある。

- 方式1のワークフローAPIだけでは，対象のアクティビティを絞り込めない場合，方式2を採用する。
  これらの考慮点を考慮して方式2を採用する場合は，以下の点に留意する必要がある。

  - ワークフローAPIの返却件数が大量であった場合，業務アプリケーションによる更なる絞り込み処理に時間がかかる。
  - 製品によっては，ワークフローAPIの返却データ形式がXML等のファイルの場合があり，その構文解析処理に多くの時間がかかる場合がある。

- 対象のアクティビティの絞り込みの際，性能面に懸念がある場合は，方式3が有効である。

例えば，画面に発明の名称等，出願に関する様々な詳細情報を一覧表示するような場合，方式1及び方式2では，ワークフローAPIでアクティビティの一覧を取得後，アクティビティ一つ一つに対して，アクティビティが持つ業務キー（出願番号等）を検索条件にしてデータベースへSQLを発行し，出願の詳細情報を取得する（一般的にBPMSがステータスを管理するデータベースと，BPMS外部のデータベースは結合して検索できないため。）。したがって，一覧表示の件数が多い場合，性能面で懸念が生じる。

また，ワークフローAPIで対象のアクティビティを十分に絞り込むことができない場合，方式2における業務アプリケーションの絞り込み処理に多くの時間がかかる可能性がある。

このように方式1及び方式2では一覧画面表示処理の性能要件を満たせない場合，方式3を採用する。

方式3を採用する場合，アプリケーションを利用してプロセスデータやステータスを個別データベースに格納する。

方式3を採用する場合は，以下の点に留意する必要がある。

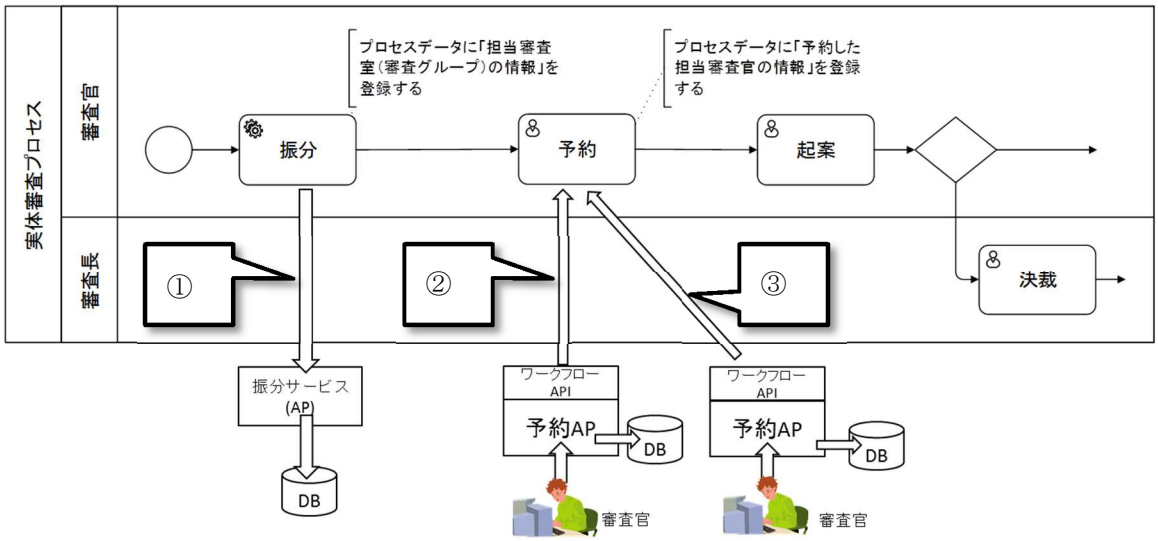
- ビジネスプロセスの変更が発生した場合には，BPMNで記載したビジネスプロセスの定義情報の変更だけなく，データベースの設計や業務アプリケーションのステータス更新部分の設計にも変更が発生する。
- 異常時，データベースとBPMS上のステータスに不整合が発生した場合のリカバリが必要である。

(C) 方式の詳細

方式1)

ワークフローAPIを使ってBPMSが保持する業務データ(プロセスデータ)を検索する。

例として,「予約対象の出願(案件)」の一覧取得の処理イメージを下図に示す。



① BPMS より, サービスを呼び出し, 対象の出願に対して, 振分け先の担当審査室(又は審査グループ)を BPMS に返却。BPMS が保持するプロセスデータに担当審査室の情報を保存。

② ユーザからの要求により, AP はワークフローAPI を使って, プロセスデータとして保存された担当審査室に対し, ユーザの属する審査室を検索条件として予約対象のアクティビティを取得。アクティビティ情報に含まれる業務キー(出願番号等)をもとにデータベースへアクセスし出願の詳細情報を取得し, 一覧画面を表示。

③ ユーザが出願の予約を要求すると, AP はワークフローAPIを使って, 業務として担当の審査官情報を送付しプロセスインスタンスを更新することで, BPMS が保持するプロセスデータに担当審査官が保存されるとともに, 起案のユーザタスクへ進む。

図 3.3-9 アクティビティに対応するヒューマンタスクの動的な絞込み(方式1)

方式2)

ワークフローAPIを使ってBPMSが保持する業務データ(プロセスデータ)を検索し, 更にプレゼンテーションロジックで対象を絞り込む。

ワークフローAPIを利用する点は方式1と同様であるが, 方式2では, ワークフローAPIが返却したアクティビティやプロセスデータの情報を, 業務アプリケーションで更に絞り込みを行う。

「予約対象の出願(案件)」の一覧取得を例にすると, 業務アプリケーションの絞り込み処理は,「図 3.3-9 アクティビティに対応するヒューマンタスクの動的な絞込み(方式1)」②の「予約AP」にて行う。

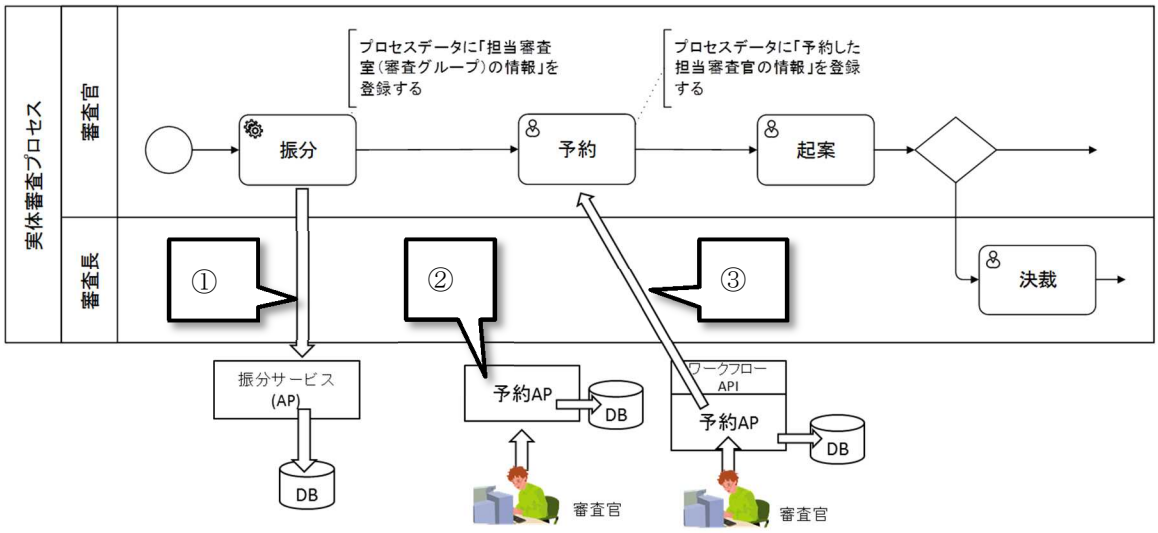
方式3)

ヒューマンタスクのステータスを個別データベースで二重管理し，アクティビティの一覧を取得するときは個別データベースのみを検索する。

BPMSと個別データベースの二重管理となり，保守性のデメリットがあるため，ヒューマンタスクのステータスに関してのみ，個別データベースで二重管理する。

一覧取得時にはBPMSにアクセスせず，個別データベースだけ検索できるようにすることで，性能要件に対処する。アクティビティを更新しプロセスを進行させる際には，方式1と同様ワークフローAPIを使ってBPMSを呼び出す。このとき，個別データベースで管理するプロセスの進捗状況も更新する。

例として，「予約対象の出願（案件）」の一覧取得の処理イメージを下図に示す。



① BPMS からサービスを呼び出し，対象の出願に対して，振分け先の担当審査室を決定。また，個別データベースに予約待ちであることを保存。
② ユーザからの要求により，AP は予約待機状態かつユーザの属する審査室を検索条件として個別データベースを検索し，対象の出願を取得。出願の詳細情報も同時に取得し，一覧画面を表示。
③ ユーザが出願の予約を要求すると，AP はワークフローAPI を使ってプロセスインスタンスを更新。個別データベースに当該ユーザを担当審査官として保存するとも起案待ちであることを保存。

図 3.3-10 アクティビティに対応するヒューマンタスクの動的な絞込み（方式3）

B.　担当者の変更運用

ユーザの実行待ちとなっているアクティビティについて，人事異動等の理由により担当者の変更をしなければならないケースがある。例えば，通常，差戻し時は起案者である担当審査官に割り当てるが，起案者が審査室外へ異動していた場合に指定分類の代表担当官に差戻したり，審査グループ案件で審査グループ外に異動していた場合に審査グループ長へ差戻したりする等，起案者とは別の審査官に差戻す必要がある。

このような場合，「A. アクティビティに対応するヒューマンタスクの動的な絞込み」で採用した方式に応じて設計内容が変わる。

- 「A. アクティビティに対応するヒューマンタスクの動的な絞込み」にて，方式1及び方式2を採用する場合には，担当者異動時，ワークフローAPIを使ってBPMSで担当者を保持するプロセスデータを更新する方式を採用する。

担当者の情報をBPMS上のプロセスデータに持つため，担当者異動時にワークフローAPIを使用して，プロセスデータの値を変更する必要があるためである。

- 「A. アクティビティに対応するヒューマンタスクの動的な絞込み」にて，方式3を採用する場合には，担当者異動時，個別データベースに保持する担当者情報を変更する方式を採用する。

担当者の情報はBPMS上のプロセスデータには保持せず，個別データベース上のみに持っているため，担当者異動時に個別データベースの値を変更する必要があるためである。

(3) BPMNの記述上想定外のイベントが発生するケース

A. ビジネスプロセスが開始する前に，中間イベントが通知されるケース

BPMNのビジネスプロセスが開始していない状態で，中間イベントが先に発生する場合がある。例えば，特許願を受理前に出願審査請求書を受理する等，存在しない出願番号が指定された中間書類を先に処理した場合が考えられる。

この場合，対象のビジネスプロセスインスタンスが存在しないため，BPMSがエラー又は誤動作してしまう。

このような事態を避けるため，以下のようにアプリケーションを利用して設計することが考えられる。

- BPMSを呼び出す前に，業務的なチェックをし，ビジネスプロセスインスタンスが存在しないと判断される場合には，BPMSへ通知しないように設計する。

例えば，出願審査請求書を受理した場合，呼び出し元のビジネスプロセス等で，事前に特許願を受理しているか（出願番号が存在するか）といった業務的なチェックを行い，存在しない場合には，BPMSを呼び出さないようにする。

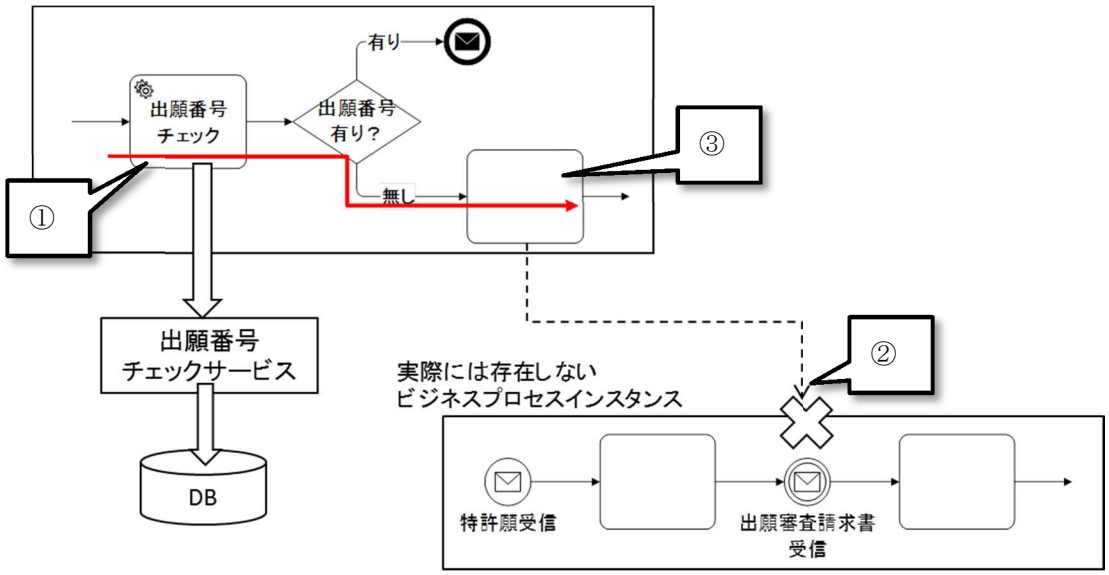
- BPMSの呼び出し元のシステム構成要素は，多階層構造図のアクセスパスにより，プレゼンテーションロジック，BPMS（ワークフローからの呼び出し），BPMS補完機能，業務アプリケーション（システム），業務アプリケーション（バッチ），ESBの6パターンがある[50]。各BPMSの呼び出し元における事前の業務チェックを行う箇所は下表に従うこと。

表 3.3-9 BPMSの呼び出し元と事前チェック箇所

項番	BPMS呼び出し元	事前チェック箇所
1	プレゼンテーションロジック	BPMS補完機能にてBPMSを呼び出す直前（事前チェックは，BPMS補完機能に委譲する）。
2	BPMS（ワークフローからの呼び出し）	
3	BPMS補完機能	BPMS補完機能にてBPMSを呼び出す直前。
4	業務アプリケーション（システム）	BPMS補完機能にてBPMSを呼び出す直前（事前チェックは，BPMS補完機能に委譲する）。
5	業務アプリケーション（バッチ）	
6	ESB	

[50] 業務アプリケーション（システム），業務アプリケーション（バッチ）は「表 3.1-8 アクセスパスの特例と許容されるアクセス条件」に示す条件でのみアクセス可能。

例として，出願審査請求書受理時のビジネスプロセスが呼び出し元だった場合の処理イメージを下図に示す。



① 出願番号が存在するか(願書を受理済か)の業務的なチェックを実施し，ビジネスプロセスインスタンスが起動していることを事前にチェック。
② 呼び出し元のチェック結果の判定により，BPMS に中間イベントは通知しない。
③ 必要に応じて，想定外のケースへ対処するフローへ進む。

図 3.3-11 ビジネスプロセス開始前での中間イベント通知防止の制御

B.　イベントゲートウェイを利用し，一方のイベントが通知されフローが進行してしまった後に，他方のイベントが通知されるケース

例えば，拒絶理由通知応答待ちについて，意見書の受理のイベントと期限経過のイベントに対するBPMNのイベントゲートウェイで表現することができる。しかし，応答期限経過後に意見書を受理した場合，期限経過のイベント発生によるフローへ進んでしまうにもかかわらず，期限ぎりぎりで受理した書類だったため受付業務実施中に応答期間経過チェックのバッチ処理が先に動いてしまうケースや，戻し止めの運用のように期間内の受理として認めるケースが想定される。

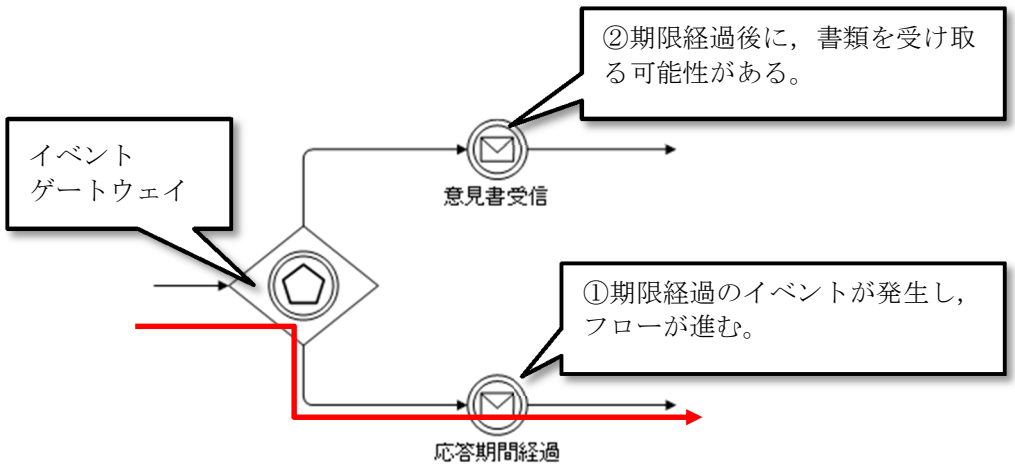


図 3.3-12 イベントゲートウェイによる排他処理

応答期限を経過した場合，BPMN上は既に期限経過のフローへ進んでしまっているため，意見書を受け取ったことによるイベントをビジネスプロセスに通知したとしても，元のフローに戻すことはできない。また，イベントが通知されてしまった場合，製品によってはエラーになったり，誤動作の原因になったりするおそれがある。

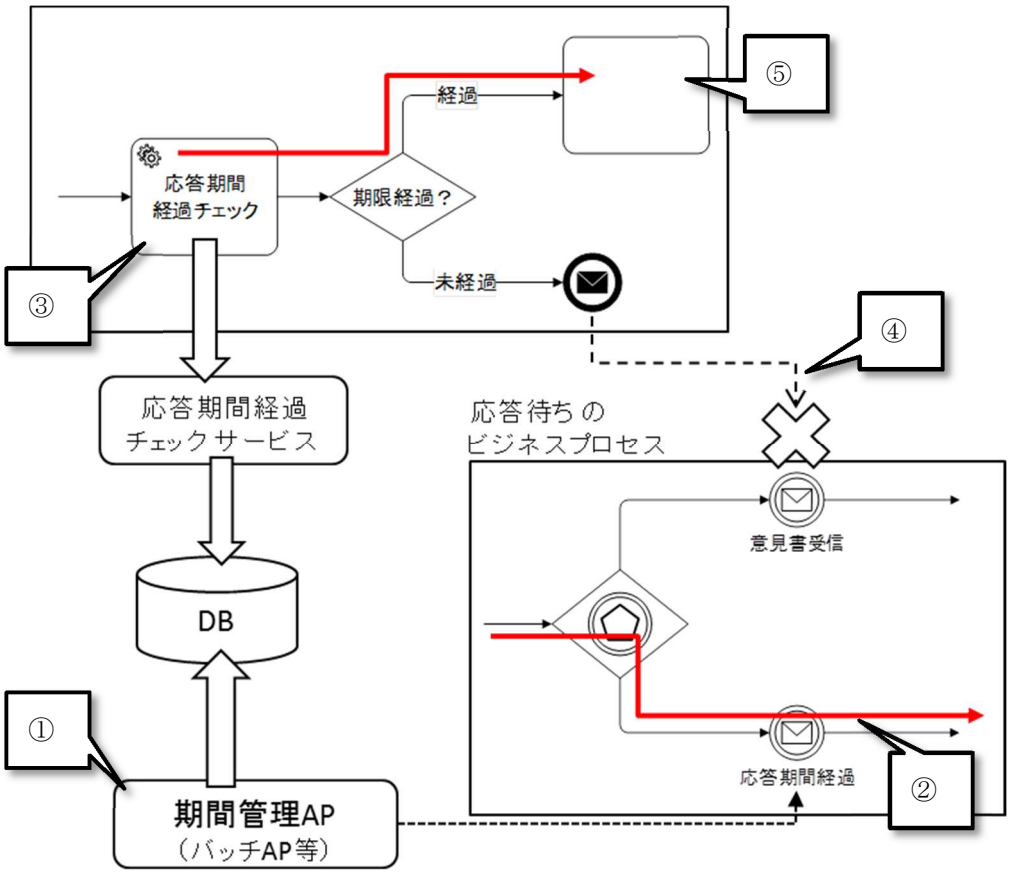
このような事態を避けるため，以下のようにアプリケーションを利用して設計することが考えられる。

- 呼び出し元のビジネスプロセス及び業務アプリケーションで，業務的なチェックを実施し，既に他方のイベントが発生しフローが進行してしまっていると判断される場合にはBPMSに通知しないようにする。

例えば，意見書受理のイベントをBPMSに通知する前に，当該出願が期限経過しているか必ず事前にチェックし，既に期限経過している場合には，BPMSに通知しないように設計する。

BPMSの呼び出し元別の事前の業務チェックの処理箇所は，「表 3.3-9 BPMSの呼び出し元と事前チェック箇所」と同様である。

例として，意見書受理時のビジネスプロセスが呼び出し元だった場合の処理イメージを下図に示す。



① データベース上の出願又は書類の情報から応答期間を計算し期限経過していた場合には，期限経過の BPMS にイベントを通知。
② 期限経過のイベントが発生し，フローが進む。
③ 書類を受理時，BPMS にイベントを通知する前に，必ず期限経過のチェックを実施。
④ 期限経過していた場合には，BPMS に通知しない。
⑤ 期限経過後に書類を受理した場合に対処するフローへ進む。

図 3.3-13 イベントゲートウェイのフロー進行後のイベント通知防止の制御

3.3.2.1.3 プロセスデータとして保持する情報

目的:	データの二重管理を防止することにより，変更による影響の局所化，拡散の抑制を図る。
スコープ:	階層定型化サブシステム
指針1:	ビジネスプロセスのプロセスデータは，BPMSではキー情報だけを保持するように設計すること。

プロセスデータとは，ビジネスプロセスインスタンスの属性値のことを指し，BPMSが自動的に生成・更新するもの（例えば，ビジネスプロセスインスタンスのID，BPMS製品の内部で管理されるデータ）はプロセスデータに含まない。「3.3.2.1.4 ビジネスプロセスの状態管理の設計指針」に示されるビジネスプロセスのステータスは，BPMS製品の内部で管理されるものであり，前述のとおりプロセスデータに含まない。

プロセスデータはビジネスプロセスをBPMSで制御する際に使用するが，データの二重管理の防止のため，BPMSではキー情報だけを持つようにする。
キー情報とは以下のものを指す。

- 業務対象を識別するID

出願番号や書類ID等，ビジネスプロセスインスタンスの業務対象を一意に示すID又はIDの組み合わせを指す。イベント通知先のビジネスプロセスインスタンスを特定するために利用される。また，BPMSが業務アプリケーション（システム）等のシステム構成要素を呼び出す際の入力項目に利用される。

- 担当者を識別するID

アクティビティ（ユーザタスク）の担当者を一意に示すID又はIDの組み合わせを指す。なお，LDAP上で定義されている組織とユーザに関する情報のうち，IDがキー情報である。

- 組織を識別するID

アクティビティ（ユーザタスク）を実行可能な組織を一意に示すID又はIDの組み合わせを指す。

- ロールを識別するID

アクティビティの実行責務を負うロールを一意に示すID又はIDの組み合わせを指す。

- ビジネスプロセスの分岐条件を判定するために必要なデータ

審査結果や決裁結果等，ビジネスプロセスが利用するサービスのレスポンス等がそれにあたる。

- 処理対象となるデータを識別するID

サブシステム間連携処理において，連携先のサブシステムに対して処理対象となるデータを特定するために通知されるものを指す。例えば，事件データを特定するための出願番号，書類データを特定するための庁内書類番号，等が該当する。

3.3.2.1.4 ビジネスプロセスの状態管理の設計指針

目的:	ビジネスプロセスの状態の管理方法，状態定義を統一することにより，変更時の影響を具体的に把握することを容易にする。また，BPMS製品のリプレース時の影響を小さくする。
スコープ:	階層定型化サブシステム
指針1:	ビジネスプロセスのステータスの管理は，「(1)状態管理のルール」に従って行うこと。
特例1:	ビジネスプロセスの状態を一覧取得する際，性能面に懸念がある場合には，「(2)ビジネスプロセスの状態の保持方法」に示す保持方法を許容する。

ビジネスプロセスの状態は，以下の2点で表される。

- ビジネスプロセスのステータス

ビジネスプロセスがどのアクティビティまで進んだかを知るためには，ビジネスプロセスの進行状況を管理する必要がある。ビジネスプロセスがどのアクティビティまで進んだかを示す情報のことを「ビジネスプロセスのステータス」と呼ぶ。

特許庁業務におけるビジネスプロセスのステータスとして，申請人又は他の課室の職員から見た大きなビジネスプロセスの進行状況（例えば，方式係属中，方式完，審査係属中，査定，審判係属中，登録係属中といった書類や出願（事件）の審査状況）や，課室内における職員レベルでの詳細なビジネスプロセスの進行状況（例えば，目視審査中，起案書決裁中といった方式審査室内の申請書類の審査状況）が定義可能だが，これらは，階層的な関係も含めBPMNによって記載することができる。BPMNでビジネスプロセスを記載することにより，BPMSがビジネスプロセスの進行状況や実行履歴を管理する仕組み（BPMSのステータス管理機能）を提供するため，システムとしてステータスの管理が可能となる。

- ヒューマンタスクの状態

ヒューマンタスクの状態とは，特定のユーザへのタスク割当済や，特定のユーザによるタスクの実施中といったBPMN上には現れないユーザタスクの詳細な状態のことを指す。

(1) 状態管理のルール

- ビジネスプロセスのステータスはBPMSのステータス管理機能を利用して管理すること。

BPMSのステータス管理機能は，ビジネスプロセスのステータスを管理する機能及びアクティビティの実行者，実行日時といった実行履歴を管理する機能を指す。

(2) ビジネスプロセスの状態の保持方法

- BPMS内部で保持しているビジネスプロセスの状態を一覧取得する際，性能面に懸念がある場合には，ビジネスプロセスの状態の一部をデータベースにも重複して保持することを許容する。ただし，以下の点に留意して利用すること。
  - ビジネスプロセスの変更が発生した場合には，BPMNで記載したビジネスプロセスの定義情報の変更だけなく，データベースの設計や業務アプリケーションのステータス更新部分の設計にも変更が発生する。
  - 異常時，データベースとBPMS上のステータスに不整合が発生した場合のリカバリが必要である。

3.3.2.2 BPMS補完機能のルール

3.3.2.2.1 BPMS補完機能で実現する機能

目的:	特定のビジネスプロセスを前提とした処理をBPMS補完機能で行い業務アプリケーションから分離することにより，業務アプリケーションとビジネスプロセスが密結合になることを防ぐ。また，BPMS補完機能の責務を統一することにより，サブシステム構造の定型化を促進し，変更時の影響箇所を具体的に把握することを容易にする。
スコープ:	階層定型化サブシステム
規約1:	BPMS補完機能で実現する機能はBPMS単体で実現できない機能に限り，BPMSと連携し補完する処理を実装すること。
規約2:	BPMS補完機能には業務アプリケーション層の責務となる業務ロジックは持たせず，ワークフロー層の責務として，起動するビジネスプロセス又は連携するビジネスプロセスインスタンスの特定やイベント通知の要否判定といった制御を行うこと。

連携するビジネスプロセスインスタンスを業務的な条件により動的に判断・特定する等，本来BPMSにて実現されるべきものであるもののBPMSの標準機能だけでは実現することが難しく，別途アプリケーションを介する必要がある場合がある。このような連携処理を実現するためにBPMSを補完する処理を行うアプリケーションがBPMS補完機能であり，ワークフロー層に配置される。

BPMSの補完処理はビジネスプロセスの定義内容に強く依存したものとなるため，ビジネスプロセスの変更に伴い，対応するBPMS補完機能の修正が必要になる可能性が高い。

BPMS補完機能が実現する機能のルールを以下に示す。

- BPMS単体で実現できない機能に限り，BPMSと連携し補完する処理を実装すること。
- 業務アプリケーション層の責務となる業務ロジックは持たせず[51]，ワークフロー層の責務として，起動するビジネスプロセス又は連携するビジネスプロセスインスタンスの特定やイベント通知の要否判定といった制御を行うこと。

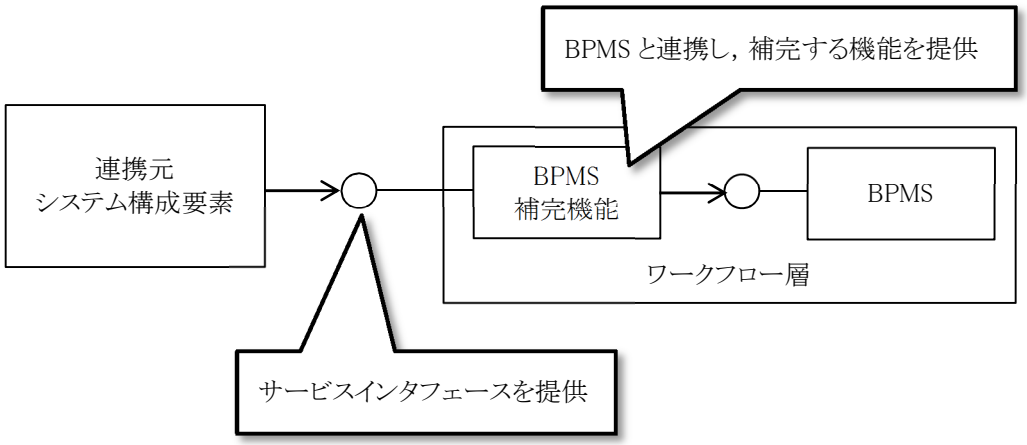


図 3.3-14 BPMS補完機能が実現する機能[52]

[51] 例えば，補完に必要な情報がデータベースにある場合，BPMS補完機能から直接データベースへアクセスは行わず，業務アプリケーション（システム）を呼び出すことにより取得する。

[52] 図中の「連携元システム構成要素」の詳細は，「3.1.1.3.1.3 システム構成要素間のアクセスパス」を参照のこと。

3.3.2.2.2 BPMS補完機能が提供する補完処理例

以下にBPMS補完機能が提供する補完処理の例を示す。

(1) 起動するビジネスプロセスやイベント通知するビジネスプロセスインスタンス及びビジネスプロセス上の対象イベントを業務的な条件で動的に特定する処理

BPMS補完機能は，連携元からの入力項目又はデータベースの格納データから，連携対象のビジネスプロセスやビジネスプロセスインスタンス及びビジネスプロセス上の対象イベントを特定し，ビジネスプロセス起動やビジネスプロセスインスタンスへイベント通知を行う。

例えば，自発補正の方式審査完了に伴う実体審査プロセスへの待機解除通知[53]は，待機解除通知するビジネスプロセスインタスタンスが静的に特定できない。このようなときにBPMS補完機能にて待機解除通知をするべきビジネスプロセスインスタンス及びビジネスプロセス上の対象イベントを特定する処理を実装する。更に，特定した単一又は複数のビジネスプロセスインスタンスに待機解除通知を行う(下図を参照)。

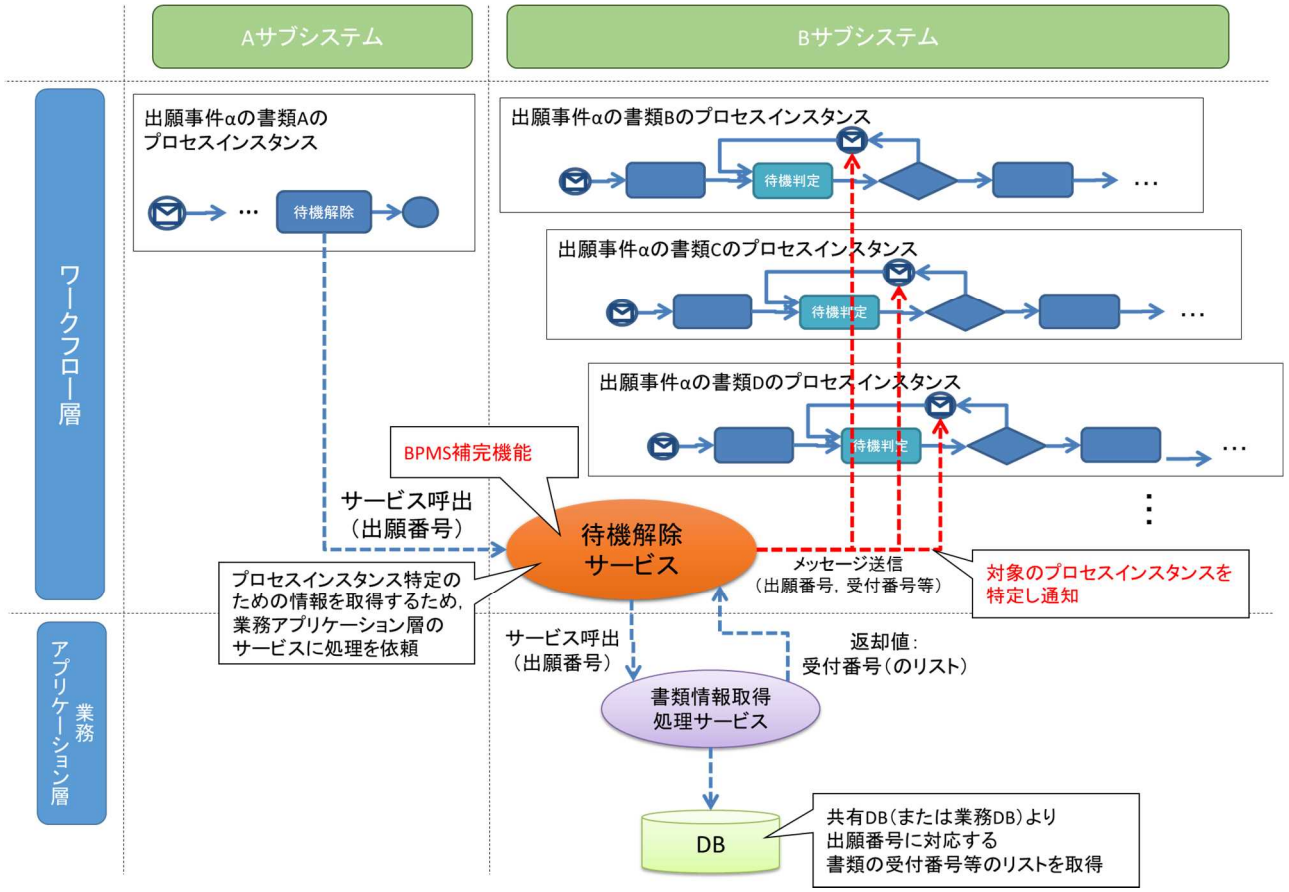


図 3.3-15 BPMS補完機能が提供する補完処理例1

[53] 本処理の詳細については，「3.3.2.1.2 BPMSの適用範囲に関する設計指針」の「(1)他のビジネスプロセスとの密接な順序関係を持つビジネスプロセス」を参照のこと。

(2)　並行運用しているビジネスプロセスの対象選定処理

『別冊2 BPMN記載ルール編』の「1.4 (1) ビジネスプロセスの記載単位ルール C.」は，制度改正により制度改正前と改正後のビジネスプロセス両方を並行運用する必要がある場合において，新旧のビジネスプロセスのうち起動させるビジネスプロセスをBPMS補完機能が選択するといった方法である。

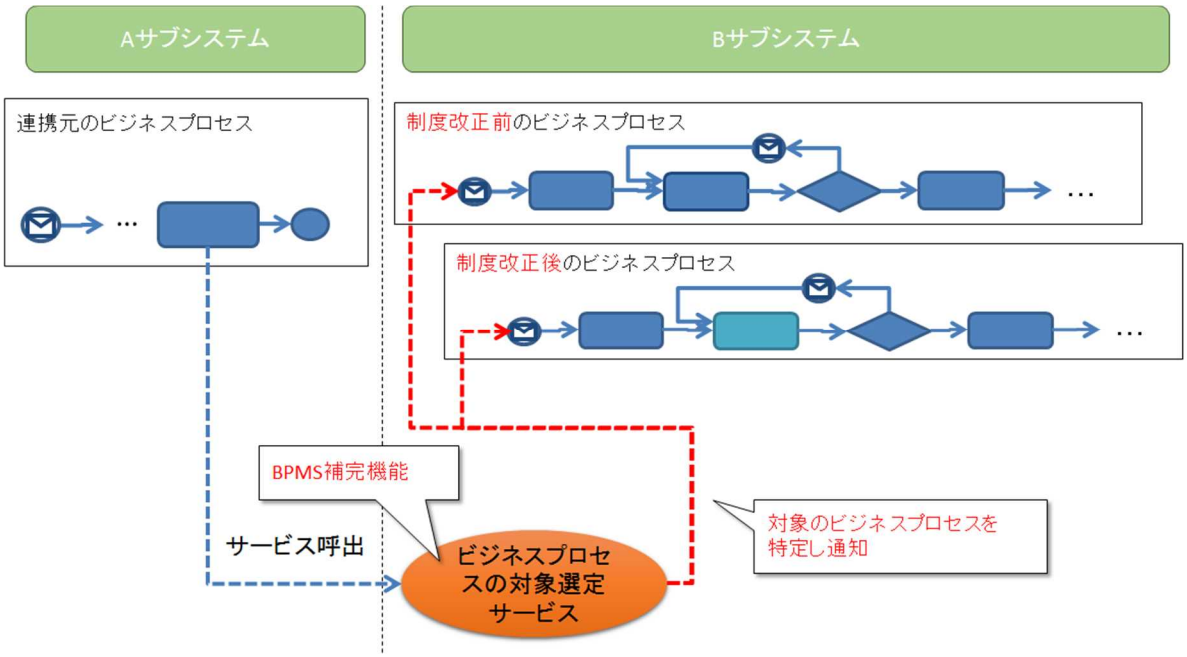


図 3.3-16 並行運用しているビジネスプロセスの対象選定処理イメージ

これにより，ビジネスプロセス内に条件分岐のためのサービスタスク及びゲートウェイを追加したり，ビジネスプロセス内に改正前後のものを併記する必要がなくなるため，ビジネスプロセスの可視性を向上させるとともに，改正前のビジネスプロセスが不要になった際に，ビジネスプロセスの修正を行うことなく破棄することが可能となる。

(3)　BPMS連携時の事前処理

通知先がイベント待ちの状態でないときにイベント通知を行った場合の挙動は製品によって異なりエラーや誤動作の原因となる。したがって，製品によってはイベント待ちの状態か否かの判定を事前に確認することが必要な場合がある。「3.3.2.1.2 (3)BPMNの記述上想定外のイベントが発生するケース」ではこのような課題に対する設計指針として，BPMSの中間イベントへの通知可否を，BPMSの呼び出し元のシステム構成要素が事前にチェックする点について定めたが，この事前チェックはBPMS補完機能が担う(「表 3.3-9 BPMSの呼び出し元と事前チェック箇所」を参照のこと)。

3.3.2.2.3 BPMS補完機能の配置位置

目的:	BPMS補完機能の配置位置を統一することにより，サブシステム構造の定型化を促進し，変更の影響を具体的に把握することを容易にする。また，利用側のサブシステムからBPMS補完機能が実現する処理を隠蔽し，サブシステム間のインタフェースの変更による影響を局所化する。
スコープ:	階層定型化サブシステム
規約1:	BPMS補完機能の配置は，補完対象のBPMSサービスを提供するサブシステムとすること。

BPMS補完機能の配置位置のルールを以下に示す[54]。

- 補完対象のBPMSサービスを提供するサブシステムに配置すること。

利用側のサブシステムからBPMS補完機能が実現する処理を隠蔽し，サブシステム間のインタフェース変更影響の局所化をするためである。

[54] 論理的な配置位置のルールであり，物理的な配置位置を定めるものではない。

3.3.3 業務アプリケーション層のルール

業務アプリケーション層の責務及び業務アプリケーション層のシステム構成要素（業務アプリケーション（ユーザ），業務アプリケーション（システム），業務アプリケーション（バッチ），個別データベース）の責務を下表に示す。業務アプリケーション層は，ビジネスロジックの実行を行うにあたり，処理方式[55]の特性に合わせた三つのアプリケーションとデータベースのシステム構成要素で構成される。

表 3.3-10 業務アプリケーション層とシステム構成要素の責務

業務アプリケーション層の責務	システム構成要素の責務			
	業務アプリケーション（ユーザ）	業務アプリケーション（システム）	業務アプリケーション（バッチ）	個別データベース
ビジネスロジック[56, 57]の実行	ユーザからの入力情報に基づく業務処理又はユーザが必要とする情報の取得	● BPMSのサービスタスクに対応し業務として意味のある一連の処理の実行 ● 他のシステム構成要素で必要とするデータの取得又は更新	BPMSのサービスタスクに対応せず所定のタイミング（期間，時間又はディレード処理要求の発生等）で処理の実行	サブシステム固有の業務データの管理

業務アプリケーション層の責務を担うために行う主な処理内容を下表に示す。

表 3.3-11 業務アプリケーション層の主な処理内容

項番	業務アプリケーション層の主な処理内容	システム構成要素			
		業務アプリケーション（ユーザ）[58]	業務アプリケーション（システム）	業務アプリケーション（バッチ）	個別データベース
1	サービスインタフェースの提供	－	○	－	－
2	プレゼンテーション層からのリクエストの受付	○	○	－	－
3	ワークフロー層からのリクエストの受付	－	○	－	－

[55] 「3.1.1.3.1.2 処理方式に基づくシステム構成要素」を参照。

[56] ビジネスロジックとは業務を遂行するための一連の処理のことである。データベースにアクセスする条件やビジネスルールの定義といった業務毎に異なるそれぞれの手続きを組み合わせることで実現する。例えば，以下の①～④の一連の処理はビジネスロジックである。

① データベースから取得したデータを変数に代入
② ①の変数を利用し，「～の場合，…とみなす」というような推論を表すビジネスルールを実行
③ ②の結果から，「～の場合，…をする」というような振分を表すビジネスルールで処理を分岐
④ 処理結果を基にデータベースを更新

ビジネスルールについては，「3.3.6.1 ビジネスルールのルール」を参照のこと。

逆に，以下のような処理はビジネスロジックではない。

- 単純な型チェック（数字，日付等）
- 単純な相関チェック（“開始日＜終了日“等）
- 入出力形式の変換（XML，CSV等の変換，フォーマット変換（日付を文字列“Hyy年mm月dd日”のように出力する等））
- HTTP等の通信処理
- 通信処理のために行う電文の解析・生成
- 複数のビジネスロジックを組み合わせて処理を行う際の呼び出し制御（例：ビジネスロジックAとビジネスロジックBを呼び出し，それぞれの結果を引数にビジネスロジックCを呼び出す）

[57] ビジネスロジックの実行は業務アプリケーション層の責務であり，業務アプリケーション層以外の層の責務には含まれない。

[58] 「図 3.1-11」にて示すとおり，特例としてUI層に配置されるものが存在する。この場合の主な処理内容は，表中の項番7，項番12が該当し，それ以外は対象外となる。

項番	業務アプリケーション層の主な処理内容	システム構成要素			
		業務アプリケーション(ユーザ)[58]	業務アプリケーション(システム)	業務アプリケーション(バッチ)	個別データベース
4	外部システム連携層からのリクエストの受付	－	○	－	－
5	ジョブ管理マネージャからのリクエストの受付又はディレード処理要求の確認	－	－	○	－
6	自サブシステムの業務アプリケーション(ユーザ),業務アプリケーション(システム),業務アプリケーション(バッチ)から,データベースがサポートするプロトコルによるリクエストの受付	－	－	－	○
7	画面とのインタラクティブなやり取りに特化したビジネスロジックの実行[59]	○	－	－	－
8	BPMSのサービスタスクに対応したビジネスロジックの実行	－	○	－	－
9	他のシステム構成要素で必要とするデータの取得又は更新[60]	－	○	－	－
10	バッチジョブに特化したビジネスロジックの実行	－	－	○	－
11	サブシステムの固有データ(個別リソースデータ及び個別業務イベントデータ)と共通リソースデータの提供	－	－	－	○
12	基盤機能層又はデータベース層に対するデータの取得又は更新の要求[61]	○	○	○	－
13	ビジネスルール管理層に対するビジネスルールの判定の要求[62]	○	○	○	－
14	外部システム連携層に対する外部システムとの連携の要求[63]	○	○	○	－
15	ワークフロー層に対するアクセス[64]	○	○	○	－

○:当該処理を担う,－:対象外

[59] 例えば,情報の検索や閲覧等。
[60] アクセスパスの制限事項に該当する場合。
[61] ビジネスロジックにデータ処理が存在する場合。
[62] ビジネスロジックにビジネスルールの判定処理が存在する場合。
[63] ビジネスロジックに外部システムへ連携する処理が存在する場合。
[64] ビジネスロジックにアクセスパスの特例に該当する処理が存在する場合。

3.3.3.1 業務アプリケーション(システム)のサービスの粒度[65]

目的:	複数のサブシステムをまたがらないようにサービスを構成することにより，サブシステム構造の定型化を図る。また，変更による影響範囲を個々のサブシステム内に局所化する。更に，サービスの配置の把握を容易にするほか，システムの運用や保守をサービスごとに独立して管理することにより運用・保守の柔軟性を向上する。
スコープ:	階層定型化サブシステム
指針1:	サービスの粒度は，「(1)サービスの粒度のルール」に従って設計すること。

業務アプリケーション層のシステム構成要素のうち，サービスを提供するのは，業務アプリケーション(システム)のみである。ここでは，業務アプリケーション(システム)が提供するサービスの粒度のルールを以下に示す。

サービスは，一つ以上のサービスインタフェースを業務のまとまりで束ねたものである。サービスインタフェースについては，「3.3.3.2 業務アプリケーション(システム)のサービスインタフェースの粒度」，サービス化の指針については，「3.2.1.1.1 サービス化の指針」を参照のこと。

(1) サービスの粒度のルール

サービスは，以下の粒度で構成すること。

- 複数のサブシステムをまたがらないように構成すること。
- サブシステム内においても，「A.サービス分割の観点」に示す観点でサービスを分割すること。

サービスの粒度のイメージを，以下に示す。

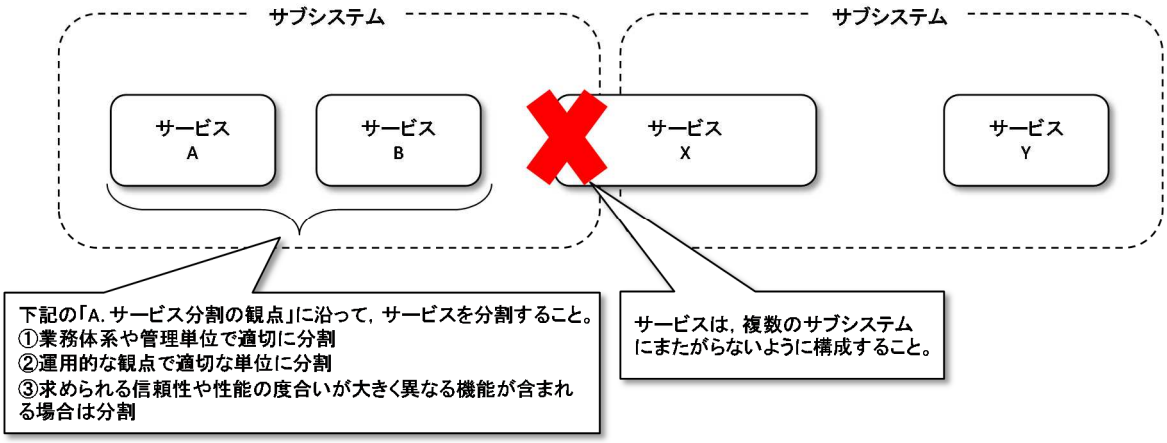


図 3.3-17 サービスの粒度のイメージ

A. サービス分割の観点

① 業務体系や管理単位で適切に分割すること。

大きなサービスの管理は人間にとって煩雑なものになる。一方，細かいサービスに分割するとサービス数が多くなり，同様に管理が煩雑になる。業務の責務の切れ目や管理単位でサービスを分割し，サービスの配置が人間にとって，直感的に把握しやすく，管理しやすい大きさにする。

② バージョンアップによるサービスの起動・停止等，運用的な観点で適切な単位で分割すること。

サービス提供時間帯が異なる等，運用上独立しているサービスは資材及びデプロイ先を分け，サービスを独立させることで，システム運用における柔軟性が向上する場合，分割を行う。

③ 求められる信頼性や性能の度合いが大きく異なる機能が含まれる場合は，サービスを分割すること。

[65] ブラウザからの要求を受けるアプリケーション(プレゼンテーションロジック及び業務アプリケーション(ユーザ)のまとまり)の粒度については，「3.3.1.4 ブラウザからの要求を受けるアプリケーションの粒度」を参照のこと。

サービスが要求するシステムリソース（CPU，メモリ量，DBコネクションプール数等）を別途与える必要があるサービス群に対しては，サービスを分割しておくこと。物理構成においても，サービスに合わせて分割することで，独立して運用が可能となる。

(2)　サービスの粒度のルールを考慮したサービス分割例

サブシステム内でサービスを分割する例としては，実体審査サブシステムにおける実体審査と検索外注業務がある。両者は，業務の責務が分かれており，開発単位や保守・運用における扱いも分けられることから，サービスは分割することが可能である。サービスを分割することで，実体審査と検索外注業務のサービス起動・停止等の運用やバージョンアップ等の保守を独立して管理することができるといったメリットがある。

3.3.3.2 業務アプリケーション(システム)のサービスインタフェースの粒度

目的:	業務的に理解できる要素(業務として意味のある最小の単位で構成されたアプリケーション)とシステム上対応するサービスインタフェースを対応付けることにより, 業務の変更に伴うシステムへの影響範囲の把握を容易にする。
スコープ:	階層定型化サブシステム
指針1:	サービスインタフェースの粒度は,「(1)サービスインタフェースの粒度のルール」に従って設計すること。

業務アプリケーション(システム)のサービスインタフェースの粒度のルールを以下に示す。

(1) サービスインタフェースの粒度のルール

- ビジネスプロセスのシステムタスクに対応してビジネスロジックを実行する業務アプリケーション(システム)のサービスインタフェースの粒度は, 業務として意味のある最小の単位とすること。

以降, システムタスクに対応した業務アプリケーション(システム)が提供するサービスインタフェースを「業務単位」のサービスインタフェースと呼ぶ。

(2) サービスインタフェースの粒度の設計時の留意点

- 業務単位のサービスインタフェースを設計する際は, 粒度のルールに従い, 必要に応じて, サービスインタフェースを統合又は分割すること。

例えば, サービスインタフェースの出力データに他のサービスインタフェースにしか利用されない制御用データが存在する場合, その二つのサービスインタフェースは依存性がある状態と判定できる。このような場合は, 粒度が小さすぎるため, サービスインタフェースの統合を行うこと。また, 環境変化によりコンポーネントの特定部分が繰り返し修正されるような場合は, サービスインタフェースに複数の業務が混在しており, 業務として意味のある最小単位になっていない可能性がある。このような場合は, サービスの分割を検討すること。分割したサービスインタフェースの連携フローはBPMSのビジネスプロセスに記述する。

- 業務アプリケーション(システム)に呼び出し元に依存した処理を実装しないこと。

例えば, 異なるアクティビティA, Bが業務アプリケーション(システム)Cのサービスインタフェースにアクセスするといった状況において, Cの実装のなかで「Aからアクセスがあった場合」,「Bからアクセスがあった場合」のような分岐処理を実装すべきではない。これは, 業務として意味のある最小の単位になっていないため, BPMSと業務アプリケーション(システム)が密結合となり, 変更が発生した場合の影響が大きくなる。仮に, アクティビティA向けの業務アプリケーション(システム)Cの変更が発生した場合, アクティビティBには関係のない内容でも変更の影響を確認する必要がある。

このように, 呼出し元に依存した処理が実装されるのはサービスインタフェースの粒度が粗いからであり, サービスインタフェースの粒度を適正にすることにより解消される(呼出し元を意識する必要がある, と言う事は, そもそもサービスインタフェースの粒度が適切に分割されていない)。当該ケースにおいてはサービスの分割を検討すること。

3.3.3.3 サブシステム間共通機能

(1) サブシステム間共通機能の定義

サブシステム間共通機能の定義を以下に示す。

サブシステム間共通機能は，サブシステム間で共通化すべきと特許庁が判断した機能とする。
以下の条件を全て満たす業務的な機能を，サブシステム間共通機能の候補とする。

- 将来的に，不変であると考える機能
- 機能の変更のときに，業務によらず，処理ロジックを一括変更すべきと考える機能
- 複数のサブシステムから利用する機能
- 以下のいずれかに該当する機能
  - 法令等で定められた重要な項目を導出する機能
  - データの整合性を保つための機能[66]

(2) サブシステム間共通機能の分類

ToBeアーキテクチャにおけるサブシステム間共通機能の分類を下表に示す。

表 3.3-12 サブシステム間共通機能の分類

項番	分類	詳細
1	サービス	詳細は，「表 3.2-2 サービス化対象」を参照のこと。
2	共有コンポーネント	● 共有ビジネスロジック（業務特有機能） ● 共有DBアクセス（基盤API） 詳細は，「3.1.2.2.2 APレイヤ・コンポーネント定義」，「3.1.2.3.2 APレイヤ・コンポーネント定義」及び「3.1.2.4.2 APレイヤ・コンポーネント定義」を参照のこと。

[66] 例えば，悲観的排他制御や楽観的排他制御といった機能のこと。

3.3.3.3.1 サブシステム間共通機能の提供形態

目的:	アプリケーション内部のコンポーネントを共有する場合のルールを定義することにより，サブシステム間の共有コンポーネントの利用を促進し，類似機能の重複開発を防止する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	サブシステム間共通機能の提供は，「(2)サブシステム間共通機能提供時の留意事項」に示す事項に留意し，提供すること。

(1) サブシステム間共通機能の提供形態

サブシステム間共通機能の提供形態を下表に示す。

表 3.3-13 サブシステム間共通機能の提供形態

項番	分類	提供形態
1	サービス	サービスインタフェースを公開する。
2	共有コンポーネント	コンポーネントをサブシステムに配付する。

(2) サブシステム間共通機能提供時の留意事項

サブシステム間共通機能を提供する際に留意すべき事項を以下に示す。

表 3.3-14 サブシステム間共通機能提供時の留意事項

項番	分類	提供時の留意事項	理由
1	共有コンポーネント	自サブシステム内の独自コンポーネントに依存しないこと。	自サブシステム内の独自コンポーネントに依存した状態であると，他サブシステムでその共有コンポーネントを利用するために独自コンポーネントまで必要になってしまうため。

3.3.3.4 個別データベースに配置するデータ及びアクセスルール

3.3.3.4.1 個別データベースの構成

目的:	個別データベースの配置及び構成するデータを統一することにより，サブシステム構造の定型化を促進し，変更時の影響を具体的に把握することを容易にする。また，個別に保有し管理すべきデータを各サブシステムに集中化することにより，変更による影響箇所の削減を図る。
スコープ:	階層定型化サブシステム
規約1:	個別データベースに配置するデータは，「(1)個別データベースに配置するデータのルール」に従うこと。

(1) 個別データベースに配置するデータのルール

個別データベースの構成ルールを以下に示す。

- 『データ統合方針書』の「2.3.2 データの具体的な配置位置の考え方」のとおり，個別データベースに配置する対象データは共通リソースデータ，個別リソースデータ及び個別業務イベントデータである。各対象データの範囲に従って設計すること。

3.3.3.4.2 個別データベースへのアクセスルール

目的:	情報活用系サブシステムによる個別データベースへのアクセスに対応する。
スコープ:	階層定型化サブシステム
指針1:	情報活用系サブシステムへのデータ提供は,「(1)情報活用系サブシステムへのデータ提供のルール」に準拠して設計すること。

(1) 情報活用系サブシステムへのデータ提供のルール

情報活用系サブシステムへデータを提供する目的で,情報活用系サブシステムから個別データベースへの直接アクセスを許容している(「3.1.1.3 多階層構造の詳細」の規約5の特例1を参照)。このアクセスにおけるルールを以下に示す。

- 個別データベースはETL製品が持つ汎用データベースアクセスインタフェース(JDBC, ODBC)に対応したライブラリを提供すること。

3.3.3.5 アプリケーション基盤機能

目的:	業務アプリケーションのコンポーネント内部の構造を定型化する事により，保守性及び移植性の向上を図る。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	ToBeアーキテクチャにおいては，「(1) ②共通部品」に示される機能を有するアプリケーション基盤（以下，「AP基盤」と称す）を構築し，これを利用すること。

(1) AP基盤とは

業務アプリケーションを構築する際，共通的に必要となる機能は，以下の二つに大別できる。

① 共通機能

サブシステム間，サブシステム内で共通的に利用する業務処理を行う機能。

② 共通部品

各機能で共通的に行う基盤系の処理を部品化したもの。業務的な意味を持たない処理。

本書におけるAP基盤とは上記の「②共通部品」を指し，OSやミドルウェアに依存するプリミティブな処理を隠蔽し，ビジネスロジックとハードウェアやミドルウェアの関係を分離するための中継機能の役割を果たす。

なお，AP基盤が持つ機能に関する具体例については，『参考3 アプリケーション基盤編』を参照すること。

(2) AP基盤の導入のルール

ToBeアーキテクチャにおいては，AP基盤を構築する事により，業務アプリケーションのコンポーネント内部の構造を定型化し，保守性及び移植性の向上を図ること。

3.3.4 外部システム連携層のルール

外部システム連携層の責務及び外部システム連携層のシステム構成要素(ESB, 外部システム互換機能)の責務を下表に示す。

表 3.3-15 外部システム連携層とシステム構成要素の責務

外部システム連携層の責務	システム構成要素の責務	
	ESB	外部システム互換機能[67]
外部システムが利用する様々な連携方式・通信方式とのギャップの吸収	ギャップの吸収における内部システムと外部システムのアクセスの一元化	ギャップのうちESBのみでは吸収できない部分の補完

外部システム連携層の責務を担うために行う主な処理内容を下表に示す。なお, 外部システム連携層にESBを含めない場合は, 表中の「ー」の部分についても外部システム互換機能が担うものとする。

表 3.3-16 外部システム連携層の主な処理内容(内部システムから外部システムへの連携)

項番	外部システム連携層の主な処理内容	システム構成要素	
		ESB	外部システム互換機能
1	サービスインタフェースの提供	○	ー
2	内部システムにおけるワークフロー層, 業務アプリケーション層からのリクエストの受付	○	ー
3	リクエストデータのデータフォーマット, 文字コード等の変換(連携先外部システムのインタフェース仕様に則った連携方式のギャップの吸収)	○	○
4	連携先外部システムのプロトコルによる外部システムへの接続, リクエスト	○	○
5	連携先外部システムからのレスポンスの受付	○	○
6	レスポンスデータのデータフォーマット, 文字コード等の変換(内部システムのインタフェース仕様に則った連携方式のギャップの吸収)	○	○
7	レスポンスデータの内部システムへの返却	○	ー

○:当該処理を担う, ー:対象外

表 3.3-17 外部システム連携層の主な処理内容(外部システムから内部システムへの連携[68])

項番	外部システム連携層の主な処理内容	システム構成要素	
		ESB	外部システム互換機能
1	サービスインタフェースの提供	○	ー
2	各外部システムがサポートする通信仕様に則ったインタフェースの提供	○	○
3	外部システムからのリクエストの受付	○	○
4	リクエストデータのデータフォーマット, 文字コード等の変換(連携先内部システムのインタフェース仕様に則った連携方式のギャップの吸収)	○	○
5	連携先内部システムのプロトコルによる内部システムへの接続, リクエスト	○	ー
6	連携先内部システムからのレスポンスの受付	○	ー
7	レスポンスデータのデータフォーマット, 文字コード等の変換(外部システムのインタフェース仕様に則った連携方式のギャップの吸収)	○	○
8	レスポンスデータの外部システムへの返却	○	○

○:当該処理を担う, ー:対象外

[67] 外部システム互換機能の実現手段(ESBのアダプタとするか, サーバ上のアプリケーションとするか等)は定めていない。

[68] ギャップがなければ, 外部システムから内部システムのサービスインタフェースに直接アクセスすることを排除するものではない。

3.3.4.1 外部システム連携層の管理ルール

3.3.4.1.1 外部システム連携層に利用する技術

目的:	アプリケーションの作り込みを極力避け，ESB製品の設定情報の変更等の対応により保守性・移植性・開発効率性を高める。また，異種のシステム間の接続に広く一般的に使用されているESBを使用することにより，ベンダロックイン排除するほか，ESBで一元化することによりシステム構造の定型化の促進を図る。
スコープ:	インタフェース定型化サブシステム（外部システム連携）
指針1:	外部システム連携層は，プログラムプロダクトを有効に活用し設計すること。
推奨1:	外部システム連携層に利用するプログラムプロダクトは，ESBを推奨する。

外部システム連携層に利用するプログラムプロダクトは，ESBを推奨する。
ESBの機能を補完する機能として外部システム互換機能を開発するものとする。

3.3.4.1.2 外部システム連携層に持たせる機能

目的:	外部インタフェースアクセスの変換機能を外部システム連携層に集中化することにより，類似の機能が各サブシステムで重複して開発されることを防ぐ。また，内部システムと外部システムを疎結合に保ち，外部システムの刷新や変更による大きな影響が内部システム側に波及することを抑止する。更に，段階的刷新によって新旧システムが混在することを考慮し，刷新により特許庁業務が阻害されないようにする。
スコープ:	インタフェース定型化サブシステム（外部システム連携）
規約1:	外部システム連携層の機能は，「表 3.3-19 外部システム連携層が持つ機能の利用可否」の利用可否に従い機能を提供すること。
特例1:	「表 3.3-19 外部システム連携層が持つ機能の利用可否」に示す特例を許容する。

目的:	外部システム連携層に連携時のギャップ吸収機能を持たせることにより，外部システムとの連携であっても標準化された内部インタフェースのようにアクセスでき，内部システムのシステム構造の定型化を促進する。また，ESBと外部システム互換機能の責務の分担を定めることにより，サブシステムの構造を定型化し，変更時の影響箇所を具体的に把握することを容易にする。
スコープ:	インタフェース定型化サブシステム（外部システム連携）
指針1:	外部システム連携層で吸収するギャップは，「(2)外部システム連携層が吸収するギャップ」に記載したものを対象とするように設計すること。
指針2:	外部システム連携層のプログラムプロダクトにESBを利用する場合，ESBと外部システム互換機能の責務の分担は，「(3)ESBと外部システム互換機能の責務の分担」に従って設計すること。

(1) 外部システム連携層の機能

外部システム連携層に基本的に必要な機能を下表に示す。

表 3.3-18 外部システム連携層が持つ機能

項番	機能名	機能概要
1	システム的なルーティング	システム的なルーティング（リクエストの内容に応じ連携先の特定を行うこと。実行すべきインタフェースの特定と実行を含む）を行う。
2	メッセージング	サービス間のメッセージの送受信を行う。また，外部システムとの連携における使用状況の確認や問題発生時の解析のためログも出力する。
3	データ変換	連携先の外部システムに合わせて，以下のようなデータ変換を行う。 ● 文字コードの変換 ● 日付のフォーマット等の変換 ● SGMLやXML形式の変換 ● コード値の変換 等
4	通信プロトコル変換	通信プロトコルを連携先の外部システムに合わせてプロトコルを変換する。
5	データフォーマットチェック	電文のデータフォーマットをチェックする。
6	リソースアクセス	共有データベースへ以下のアクセスを行う。 ● 外部システムに連携する元データを共有データベースから取得する。 ● 外部システムから連携されたデータを保存する。
7	アダプタ	パッケージ製品固有のプロトコルやホストと接続する。

なお，刷新過渡期のToBe対象システムとの連携においてのみ利用を制限する機能もある。これらの機能は，刷新後，他の機能等で実現する。

表 3.3-19 外部システム連携層が持つ機能の利用可否

項番	機能名		外部システム連携層が持つ機能の利用可否 （利用可：○，利用不可：×）	備考
1	システム的なルーティング		○	
2	メッセージング	メッセージの送受信	○	
3		システム間連携ログ	○	
4		送達保証 （再送，重複排除）	× 理由：多重でアクセスした場合に性能低下の懸念があるため。	
5	データ変換		○	詳細は「(2)外部システム連携層が吸収するギャップ」を参照のこと。
6	通信プロトコル変換			
7	データフォーマットチェック		外部システムが刷新過渡期のToBe対象システムである場合 × 理由：外部システムが内部システムになる際に外部システム連携層が不要となり，機能の移行が必要となるため。 外部システムが上記以外の場合 ○	外部システム連携層を介した外部システムからの受け口である内部システム側のサービスに，当該機能を持たせておくことが望ましい。
8	リソースアクセス		○	
9	アダプタ		× 理由：ベンダロックインを極力排除するため。	外部システムとの連携に必要な場合に限り，利用することを許容する（※特例）。

(2) 外部システム連携層が吸収するギャップ

外部システム連携層が吸収するギャップを下表に示す。なお，内部システムと外部システムの連携で表中のギャップが発生しない場合は外部システム連携層の設計に当該ギャップを含める必要はない。また，システム固有の要件により表中のギャップ以外のギャップを外部システム連携層で吸収する必要があると考えられる場合は，特許庁と協議の上決定する。

表 3.3-20 外部システム連携層が吸収するギャップ

項番	ギャップ	吸収するギャップの内容
1	プロトコル	内部システムはアーキテクチャ標準仕様で定めるプロトコルを利用するが，外部システムでは多様なプロトコルが利用されるため，プロトコルの変換を行う。 内部システムのプロトコルの詳細は，「3.2.1.1.2 内部インタフェースのプロトコル」を参照のこと。
2	文字コード	内部システムはアーキテクチャ標準仕様で定める文字コードを利用するが，外部システムでは多様な文字コードが利用されるため，文字コードの変換を行う。 内部システムの文字コードの詳細は，「3.4.2 文字コードの扱いのルール」を参照のこと。
3	データフォーマット	内部システムはアーキテクチャ標準仕様で定めるデータフォーマットを利用するが，外部システムでは多様なデータフォーマットが利用されるため，データフォーマットの変換を行う。 内部システムのデータフォーマットの詳細は，「3.2.1.1.2 内部インタフェースのプロトコル」を参照のこと。
4	連携データ	内部システムと外部システム連携層の間の連携データは，外部システム刷新後の連携方式がワークフロー連携になる場合にギャップが発生する。刷新後にワークフロー連携が想定される外部システムと連携する場合，内部システムとESBとの通信には基本的にプロセスデータのみを受け渡し，その他の連携データは共有データベースを介して受け渡しする。 外部システムとデータをまとめて受け渡しする場合は，外部システム連携層が連携データの分解又は統合を行う。
5	連携タイミング	内部システムで扱うデータは，基本的にリアルタイム連携による業務単件データである。 外部システムと連携するにあたり，業務単件データをまとめて連携する場合，外部システム連携層では以下の処理を行う。 ● 内部システムから外部システムへ連携する場合 外部システム連携層内に一時的に連携データを保持しておき，処理単位毎（時間等）に連携データを統合して連携する。[69] ● 外部システムから内部システムへ連携する場合 外部システム連携層で連携データを単件単位に分割し，業務単件データ毎に順次，内部システムと連携する。
6	通信データ量	内部システムは連携時に受け渡すデータ量が小さい（基本的に連携データはデータベースに格納する）。 外部システムは連携時に受け渡すデータ量が大きい（連携に必要なデータを全て連携時の引数で受け渡す）。また，大量データの連携がある。 そのため，外部システム連携層では以下の処理を行う。 ● 内部システムから外部システムへの連携する場合 共有データベースに格納されている連携データを統合し，外部システムに送信する大量データを生成する。 ● 外部システムから内部システムへ連携する場合 外部システムから受信した大量データを共有データベースへ格納する。

[69] 外部システム連携層から外部システムに連携を行う際の実行の契機は，運用監視システムが担ってもよい。

(3) ESBと外部システム互換機能の責務の分担

外部システム連携層で吸収するギャップに対して，ESBと外部システム互換機能のそれぞれの責務を下表に示す。

表 3.3-21 ESBと外部システム互換機能の責務

項番	ギャップ	ESB	外部システム互換機能
1	プロトコル	採用するESB製品が保有する機能で対応できるギャップの吸収処理を行う。	採用するESB製品のアダプタが対応していないプロトコルの受送信の機能を提供する。
2	文字コード		採用するESB製品が対応していない文字コードの変換機能を提供する。
3	データフォーマット		採用するESB製品の機能だけでは変換できないデータフォーマットの変換機能を提供する。
4	連携データ		採用するESB製品の機能だけでは過不足を補完できない連携データの補完機能を提供する。
5	連携タイミング	—	連携タイミングのギャップに対応する機能を提供する。
6	通信データ量	—	通信データ量のギャップに対応する機能を提供する。

3.3.5 基盤機能層及びデータベース層のルール

基盤機能層とデータベース層の責務及びシステム構成要素を以下に示す。なお，本節のシステム構成要素を持つ共有データベース管理サブシステムの外部仕様及び利用方法等については，『DBアクセス基盤利用者向けガイドライン』に従うこと。

(1) 基盤機能層

基盤機能層の責務及び基盤機能層のシステム構成要素（DBアクセス基盤サービス）の責務を下表に示す。

表 3.3-22 基盤機能層とシステム構成要素の責務

基盤機能層の責務	システム構成要素の責務
	DBアクセス基盤サービス
データベース層へのアクセス	共有データベースにアクセスするためのサービスの提供

基盤機能層の責務を担うために行う主な処理内容を以下に示す。

- サービスインタフェースの提供
- 業務アプリケーション層又は外部システム連携層からのリクエストの受付
- データベース層に対するデータ取得又は更新の要求
- データベース層からのレスポンスデータの編集[70]
- 業務アプリケーション層又は外部システム連携層へのレスポンスデータの返却

(2) データベース層

データベース層の責務及びデータベース層のシステム構成要素（共有データベース）の責務を下表に示す。

表 3.3-23 データベース層とシステム構成要素の責務

データベース層の責務	システム構成要素の責務
	共有データベース
特許庁システム全体で共有すべきデータの管理	特許庁システム全体で共有すべきデータの管理

データベース層の責務を担うために行う主な処理内容を以下に示す。

- 汎用データベースアクセスインタフェースを提供する。
- 業務アプリケーション層，基盤機能層又は外部システム連携層から，データベースがサポートするプロトコルによるリクエストの受付
- 事件・書類データ及び特定サブシステム間共有データ[71]の提供

[70] 例えば，未公開データを取得できないようにしたり，秘匿情報をマスキングするような処理は，業務によって異なる処理であり，ビジネスロジックに相当する。そのため，基盤機能層の責務には含まれない。

[71] 詳細は，『データ統合方針書』の「2.3 データ配置の考え方」を参照のこと。

3.3.5.1 共有データベースに配置するデータ及びアクセスルール

目的:	事件及び書類データの単位で一元管理することにより，大きな業務の区切り毎に共有データベースを更新することが可能となり，更新整合性を確保しやすくする。また，共有データベースへのアクセスルールを統一することにより，システム構成要素同士の複雑な連携を排除し疎結合に保つほか，サブシステム構造の定型化を促進し，変更時の影響箇所を具体的に把握することを容易にする。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	共有データベースに配置するデータは，「(1)共有データベースに配置するデータのルール」に従うこと。
規約2:	共有データベースへのアクセスは，「(2)共有データベースへのアクセスルール」に準拠すること。
指針1:	情報活用系サブシステムへのデータ提供は，「(3)情報活用系サブシステムへのデータ提供のルール」に準拠して設計すること。

(1) 共有データベースに配置するデータのルール

共有データベースの構成ルールを以下に示す。

- 『データ統合方針書』の「2.3.2 データの具体的な配置位置の考え方」のとおり，共有データベースに配置する対象データは事件・書類データ，特定サブシステム間共有データである。各対象データの範囲に従って設計すること。

(2) 共有データベースへのアクセスルール

共有データベースへのアクセスルールを以下に示す。

- 以下のシステム構成要素からのアクセスのみを許可する。
  - 業務アプリケーション(ユーザ)
  - 業務アプリケーション(システム)
  - 業務アプリケーション(バッチ)
  - DBアクセス基盤サービス
  - 外部システム互換機能
- 下表に示す基盤APIを使用する。

表 3.3-24 基盤APIと使用条件

項番	基盤API	説明	使用条件
1	基盤API(サービス版)	DBアクセス基盤サービスを介してアクセスを介してアクセスする。	● データを取得する場合 ● ートランザクションに閉じた更新を行う場合
2	基盤API(データベース版)	直接共有データベースにアクセスする。	● 基盤API(サービス版)で用意する単位をまたぐ(複数を一括でまとめる)単位で，更新を含む業務上のトランザクションを必要とする場合 ● 基盤API(サービス版)を使用することにより，要求性能が満たせない場合

- 特定サブシステム間共有データは，上記までのルールに加え，データの授受を行う，送信元サブシステムと送信先サブシステムのみアクセス可能とすること。

なお，共有データベースへのアクセスルールの詳細は，『DBアクセス基盤　利用者向けガイドライン』に従うものとする。

(3)　情報活用系サブシステムへのデータ提供のルール

情報活用系サブシステムへデータを提供する目的で，情報活用系サブシステムから共有データベースへの直接アクセスを行う場合におけるルールを以下に示す。

- 共有データベースはETL製品が持つ汎用データベースアクセスインタフェース（JDBC，ODBC）に対応したライブラリを提供すること。

3.3.6 ビジネスルール管理層のルール

ビジネスルール管理層の責務及びビジネスルール管理層のシステム構成要素(BRMS)の責務を下表に示す。

表 3.3-25 ビジネスルール管理層とシステム構成要素の責務

ビジネスルール管理層の責務	システム構成要素の責務
	BRMS
ビジネスルールの管理, 実行	ビジネスルールの管理, 実行

ビジネスルール管理層の責務を担うために行う主な処理内容を以下に示す。

- サービスインタフェースの提供
- ビジネスルールの提供
- 業務アプリケーション層からのリクエストの受付
- ビジネスルールの実行
- ビジネスルールの実行結果の返却

3.3.6.1 ビジネスルールのルール

ビジネスルールとは，手続的な処理は持たず実行順序によらない宣言的な記述が可能なものである。ビジネスルールには，以下のようなものがある。

- 「～の場合，…とみなす」というような推論[72]を表すもの
- 「～から，…を算出する」というような計算[73]を表すもの
- 「～の場合，…をする」というような振分[74]を表すもの
- 「～の場合のみ…できる」というような制約[75]を表すもの

データベースやファイルへのアクセス，ネットワークを介した他サブシステムへアクセス等，システムリソースを扱う処理はビジネスルールの実現上必要な手続きであり，ビジネスルールではない。保守性の観点から原則，ビジネスルールとは区別して管理する。

ただし，ごく簡単なビジネスルール（計算式等）とデータベースアクセス等が入り混じった（そのような形にせざるを得ない）手続きも存在し得る。このような手続きに関して無理にビジネスルールとデータベースアクセス等を分けて管理するとかえって保守性が悪くなる（規模の増加や解析性が低下する）ため，同一の実装手段で統一するものとする。

ビジネスルールを組み合わせた手続的な処理をビジネスロジックという。ビジネスロジックの詳細は，「表 3.3-10 業務アプリケーション層とシステム構成要素の責務」の「脚注56」を参照のこと。

「振分」，「制約」のビジネスルールについては，フロー図での分岐で表現されるもので，ビジネスロジックで扱うレベルのビジネスルールであれば，処理の流れで出現する分岐（プログラムコードであればif文）で表現されるものである。一方，ビジネスプロセスで扱うレベルの「振分」，「制約」のビジネスルールは，業務フロー図の分岐（BPMNであれば排他ゲートウェイ）として表現され，BPMSで管理されるものである。

ビジネスルールの条件部又は判定部（例えば，「～の場合，…とみなす」の「～」又は「…」の部分）には，数値，文字列，日付等の固定値が現れる。これらをパラメータと呼び，以下の特徴を持つ。

- ビジネスルール管理層，業務アプリケーション層だけでなく，プレゼンテーション層，ワークフロー層等業務アプリケーション全体に現れる。
- 業務処理の実行によって変更されず，運用業者又は保守業者によって変更される。
- 数値，文字列，日付等の単純な値や，それらの組み合わせがパラメータの候補となる。

---

[72] 特定の条件が真のときに新しい知識を発見するルール。例えば，年齢が「6」以下の場合は幼児，「7」以上「12」以下の場合は小学生，「13」以上「15」以下の場合は中学生，「16」以上の場合は義務教育終了とみなす。

[73] 特定の数式やアルゴリズムを使用した計算のルール。例えば，就学区分と年齢，学生証の有無から，宿泊料金を算出する。

[74] 特定の状況下で何らかのアクションを引き起こすといったルール。例えば，就学区分が幼児の場合，優先券の発券をする。

[75] システムやそのユーザーが実行するかもしれないアクションを制限するルール。例えば，空席有りの場合だけが，申込み登録処理を実施できる。

ビジネスルールと，ビジネスロジック，ビジネスプロセス，パラメータの関係を概念的に図示化したものを以下に示す。

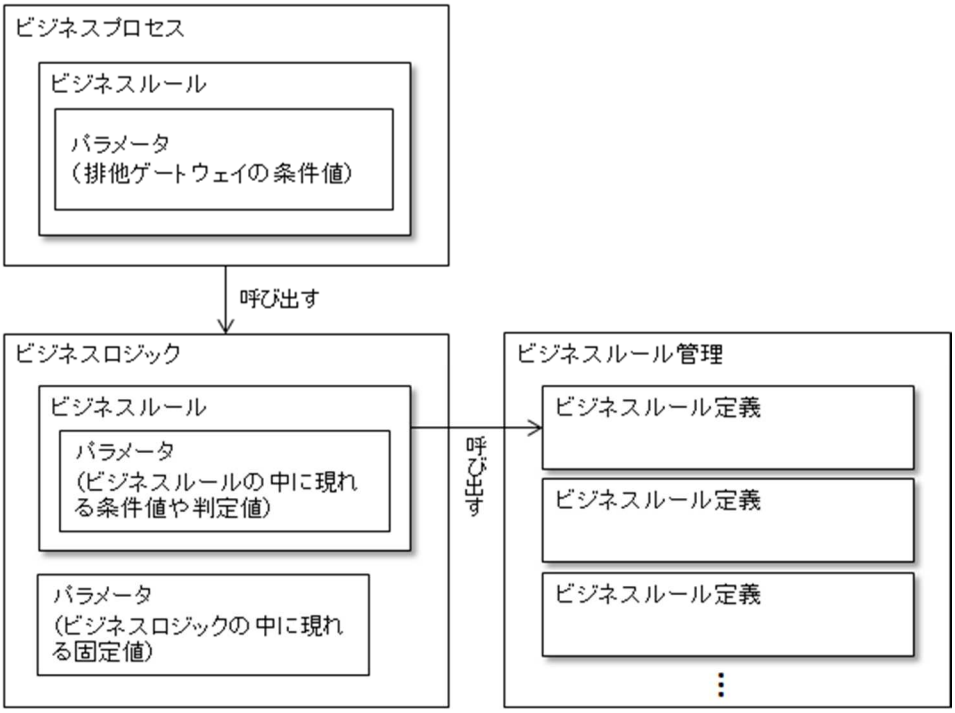


図 3.3-18 ビジネスルールと，ビジネスロジック，ビジネスプロセス，パラメータの関係

3.3.6.1.1 ビジネスルールをアプリケーションから切り離すための仕組みとして利用する技術

目的:	変更頻度が高い業務をビジネスルールとして業務アプリケーション又はBPMSから切り出し，疎結合とすることで，ビジネスルールの変更や追加の影響を局所化する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム(ビジネスルール管理)
規約1:	ビジネスルールを業務アプリケーション又はBPMSから切り離すための仕組みは，BRMSを用いること。

目的:	ビジネスルールの記述ルールや適用対象を統一することにより，ビジネスルールの可読性を高め，変更による影響箇所の特定を容易にする。また，コンポーネントレベルの依存関係を制御し，ビジネスルールを疎結合にすることにより，変更による影響範囲を局所化する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム(ビジネスルール管理)
規約1:	BRMSで使用するデータは，「(3)BRMSで使用するデータ取得に関するルール」に従うこと。
指針1:	BRMSの適用は，「(2)BRMSを用いたビジネスルールの記述対象」に従って設計すること。
特例1:	ビジネスルールは，「(2)BRMSを用いたビジネスルールの記述対象」に示す特例を許容する。

(1) ビジネスルールを業務アプリケーション又はBPMSから切り離すための仕組み

ビジネスルールを業務アプリケーション又はBPMSから切り離すための仕組みとして「BRMS」を利用する。

(2) BRMSを用いたビジネスルールの記述対象

入力チェック処理における，入力データ単体で完結するチェック処理(データの性質に基づくチェック等)，あるいは入力データ間の相関関係に基づく単純なチェック処理のようなものにBRMSを利用すると，ビジネスロジックの手続的な処理部分の設計情報とビジネスルールの設計情報がばらばらに管理されることにより可読性が低下する。また，複数のビジネスロジックから利用されないものや，変更頻度が低いビジネスルールについてBRMSを利用しても保守性のメリットが小さい。そのため，以下の二つの特性を併せ持つビジネスルールを対象に「推論」，「計算」のビジネスルールを記述する。

- 複数の「定型的な判定条件群と，結果」の組み合わせが多数現れるビジネスルール[76]
- 変更が発生しやすいビジネスルール又は複数箇所から利用されるビジネスルール

ただし，以下のケースに該当する「推論」，「計算」のビジネスルールにおいては，特例を許容する。

- ビジネスルール(計算式等)とデータベースアクセス等が入り混じった手続きに関して，無理にビジネスルールとデータベースアクセス等を分けて管理するとかえって保守性が悪くなる場合，これらを合わせてビジネスロジックに記述することを許容する。

「振分」，「制約」のビジネスルールについては，業務アプリケーションやBPMSで記述するものとする。

- 業務アプリケーション

ビジネスロジックレベルの「振分」及び「制約」は，プログラムコード上の分岐(if文等)で記載する。

特例として，BRMS製品がルールのフロー制御機能を持っている場合，ルールの可視性を低下せず，各ルールの再利用性が妨げられない範囲においてのみ，フロー制御機能の使用を許容する。

- BPMS

ビジネスプロセスレベルの「振分」及び「制約」は，BPMNの排他ゲートウェイで記載する。

排他ゲートウェイの詳細は『別冊2 BPMN記載ルール編』を参照のこと。

[76] このようなビジネスルールは，機能をテーブル形式で定義すると分かりやすく，又管理しやすいものとなる。

BRMS及び業務アプリケーション，BPMSの使い分けのイメージを下図に示す。

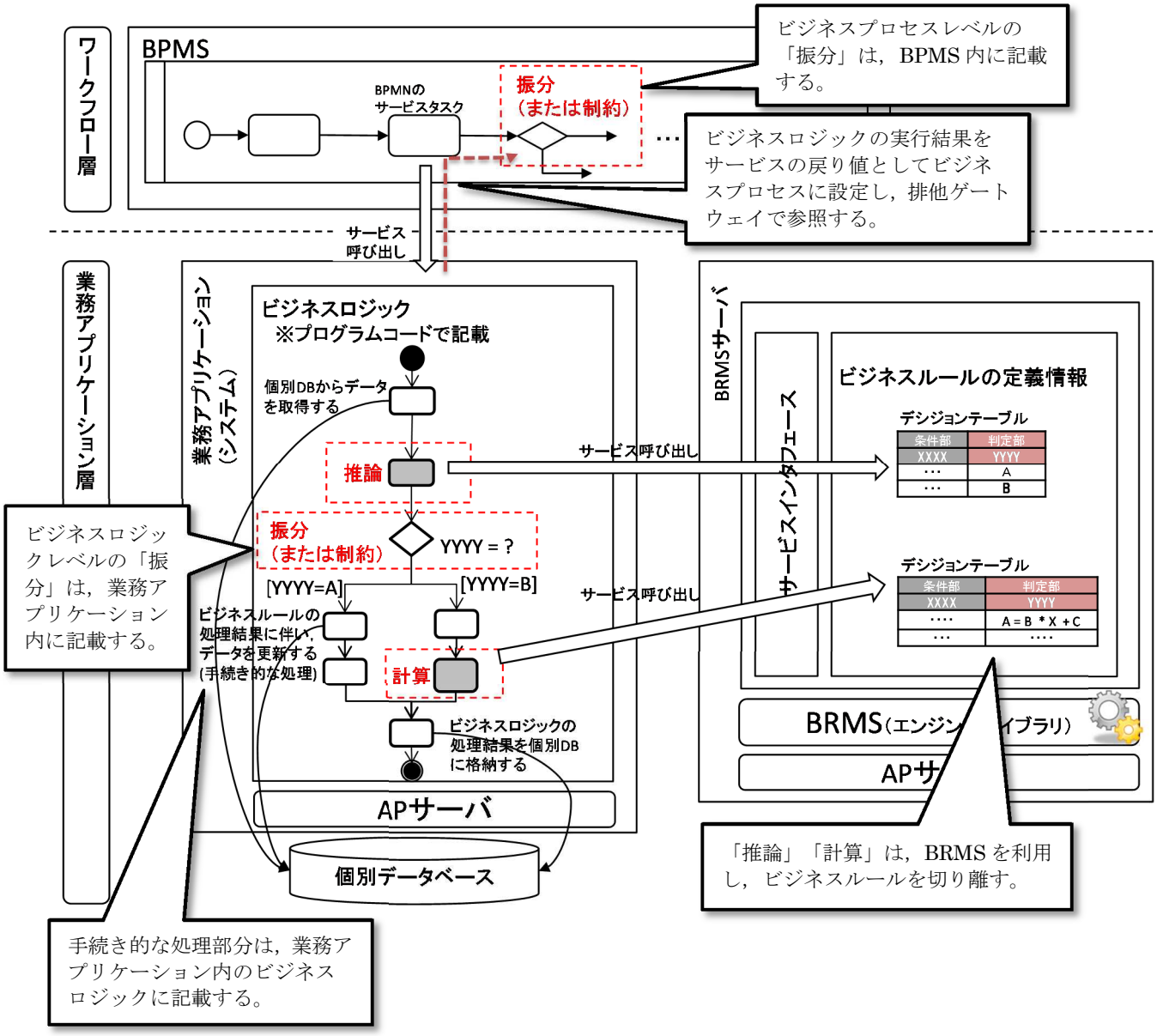


図 3.3-19 BRMS及び業務アプリケーション，BPMSの使い分けのイメージ

(3) BRMSで使用するデータ取得に関するルール

BRMSで使用するデータ取得に関するルールを以下に示す。

- BRMSから直接データベースにアクセスしデータを取得することを禁止する。データベースからのデータ取得は，BRMSの呼び出し元である業務アプリケーションで行い，BRMSへ渡すこと。

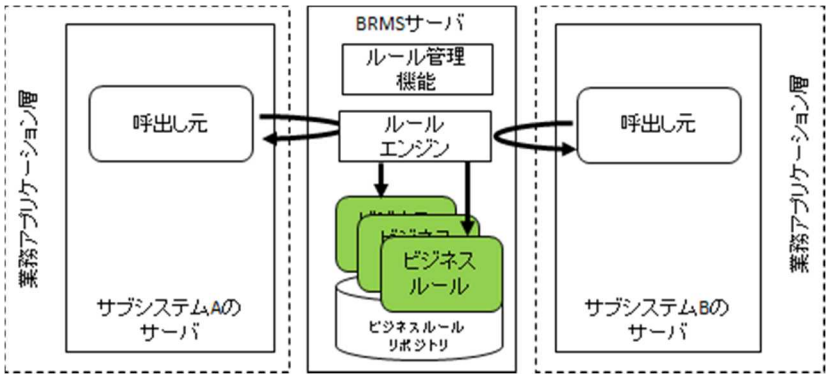
3.3.6.1.2 BRMSの呼び出し方式

目的:	ビジネスルールの実行をサービスとして提供することにより，インタフェースが変わらない限り，サービスの内部構造が刷新等により変更された場合においても，その影響が利用者側に波及しないようにする。また，BRMSの機能を集約することにより，改造が必要になった場合に影響箇所の特定を容易にし，サブシステムの移植性を向上させる。
スコープ:	インタフェース定型化サブシステム
規約1:	BRMSの呼び出し方式は，「サービス方式」とすること。

(1) BRMSの呼び出し方式

BRMSの呼び出し方式を下表に示す。

表 3.3-26 BRMSの呼び出し方式

項番	呼び出し方式	説明
1	サービス方式	ビジネスルールの実行をサービスとして提供する。 多階層構造上，BRMSはインタフェース定型化サブシステムに位置づける。 

3.4 システム開発全般で遵守すべきルール

3.4.1 データ設計に関するルール

3.4.1.1 XMLとSGMLのデータの扱いのルール

目的:	XMLとSGMLの形式の違いを吸収する階層の範囲を定め，XML形式とSGML形式を扱うコンポーネントを別にすることにより，SGML形式に依存した処理を疎結合にして，段階的刷新によって将来不要になった場合の影響範囲を局所化する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
指針1:	XMLやSGMLの書類形式を意識した業務処理は，「図 3.4-1 書類形式を意識する層の範囲」に示す範囲に配置するよう設計すること。
特例1:	書類形式を保持したまま，画面での処理を行う業務要件[77]がある場合には，UI層，プレゼンテーション層での処理を許容する。

特許庁システムで扱う書類形式は，XML及びSGMLがあるが，SGMLの書類は，将来XML化する方針である。ただし，既に蓄積済みのSGMLはXML化せず，そのまま用いる可能性がある。文書形式の移行方針については，『特許庁システム移行方針書』を参照のこと。

ToBeアーキテクチャにおいては，以下に示すXML及びSGMLのデータの扱いのルールに従うこと。

- XMLやSGMLの書類形式を意識した業務処理（書類形式の判定処理，スキーマチェック，パース処理等）は，多階層構造における以下の層に配置し，上位層に対して書類形式を隠蔽すること。
  - 業務アプリケーション層
  - 基盤機能層
  - データベース層
  - 外部システム連携層（外部システムと内部システムとの連携時に扱う書類形式が異なる場合のみ）
  - ビジネスルール管理層

ただし，書類形式を保持したまま，画面での処理を行う業務要件がある場合には，UI層，プレゼンテーション層での処理を許容する。

[77] 例えば，SGMLやXMLのタグ構造も含めて画面に表示させるような機能，等。

書類形式を意識する層の範囲を下図に示す。

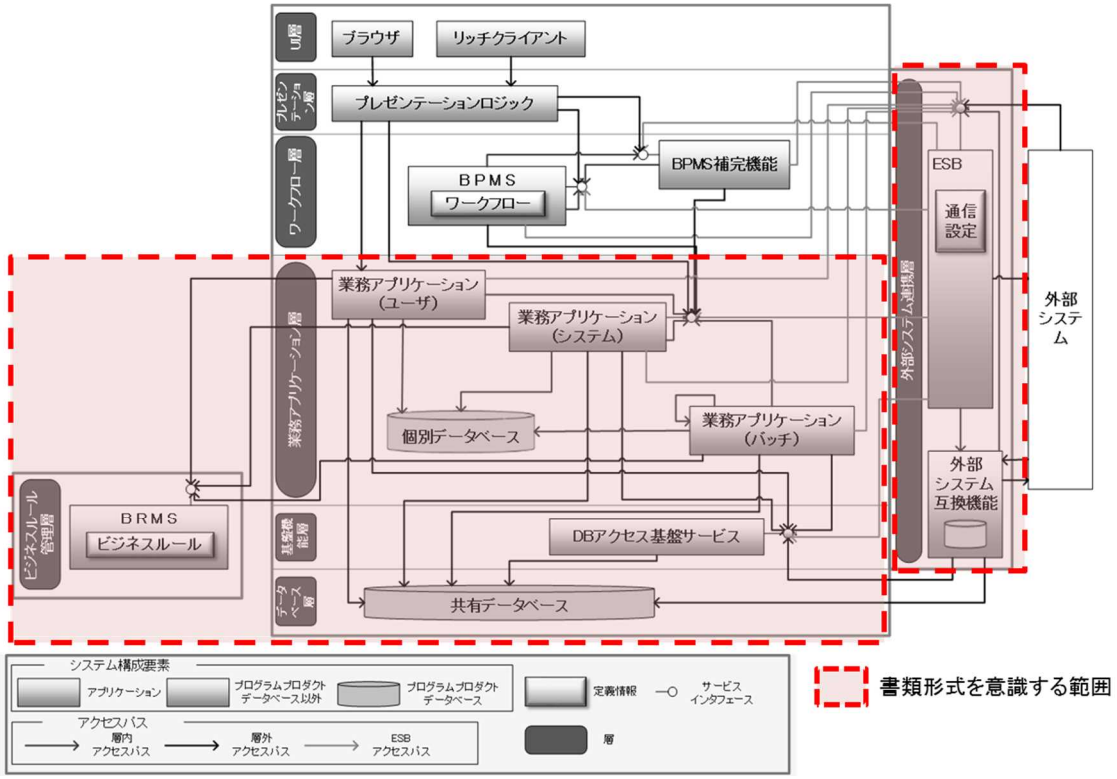


図 3.4-1 書類形式を意識する層の範囲

3.4.1.2 処分が決定した事件データの扱いのルール

目的:	処分決定後の事件データの取り扱いを定めることにより，データ量増加に伴うレスポンス低下及び維持管理コストの増加を抑止する。また，アプリケーションからデータへのアクセス方法の定型化を促進し，開発効率化を図る。
スコープ:	階層定型化サブシステム（共有データベース管理）
指針1:	処分が決定した事件データは，「表 3.4-1 処分が決定した事件データの扱い方針」に示す方針に従い設計すること。

利用頻度が低い又は全く利用しないデータを，未整理のままデータベースで管理することによって，データ量の増加に伴うレスポンスの低下や維持管理コストの増加を招くおそれがあるため，それらのデータを適切に管理するためのルールを下表に定める。

表 3.4-1 処分が決定した事件データの扱い方針

項番	要件	方針
1	処分が決定した事件データの扱い	● データの種類[78]毎に保存期間を設けること。 ● 保存期間を超過したデータをデータベースから削除する仕組みを用意すること。 ● 事件の権利消滅後の保存期間内において，当該事件に対して遡及手続されることを考慮し，アプリケーションからのデータアクセス方法を権利消滅前のアクセス方法と変えないこと。

なお，処分が決定した事件データとは，例えば以下のものを指す。

- 最終処分が決定した事件のデータ
  - 権利の有効期限が過ぎた事件のデータ。
- 関連する制度が終了したデータ
  - 「出願公告制度」に関するデータ（平成8年に廃止）等。

[78] 『データ統合方針書』におけるエンティティのカテゴライズのことを指す。詳しくは，『データ統合方針書』の「2.1.2.1 エンティティのカテゴリ定義」を参照すること。

3.4.2 文字コードの扱いのルール

(1) 文字コードの扱いのルールの方針

文字コードの扱いのルールにおける方針を以下に示す。

なお，文字コードは文字の集合である「文字セット」(符号化文字集合)とバイトコードのマッピングを示した「エンコーディング」(符号化方式)の二つを指す。

- ToBe対象システムでは，保守性を考慮し，内部システムで扱う文字コードを統一する。
- ToBe対象システムが扱っている文字は，内部システムでも利用できるようにする。
- 外部システムに極力，改修が発生しないように考慮する。
- ToBeアーキテクチャが長期にわたって適用されることを想定し，将来的に必要となると考えられる文字セットも扱えるようにする。

(2) ToBe対象システムが使用する文字

ToBe対象システムが使用する文字は以下に示す規定内文字及び特許庁外字に分類される。

A. 規定内文字

特許庁システムの文書仕様[79]における文字種の規定に基づき，以下を規定内文字とする。

- JIS X 0201-1976(1byte)
- JIS X 0208-1997(2byte)

ただし，半角カナ，丸文字，外字を除く。

B. 特許庁外字

規定内文字以外で，特定の文字を独自に規定し，特許庁システムにおいて運用を行っている文字種を「特許庁外字」とする。

特許庁外字の分類を下表に示す。

表 3.4-2 特許庁外字の分類

項番	外字分類	説明	発生起因
1	システム独自要件に基づく文字	特許庁システムにおいて，サブシステム独自に利用要件があり，個別に登録又は特別な処理を行っている文字種	特許庁内要望
2	マドプロ用のラテン文字	IBから送付されるマドプロ出願に含まれるラテン文字	特許庁外要望
3	人名・地名用の外字	職員等の姓名及び地名表示のために個別登録している外字	事務的要望

上記を踏まえ，ToBe対象システムが使用する文字の構成を下図に示す。

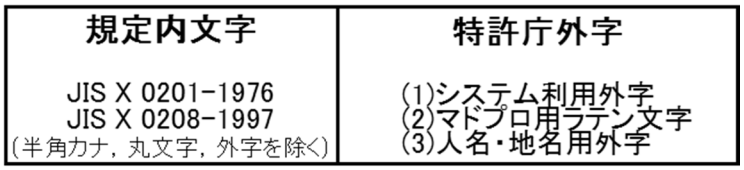


図 3.4-2 ToBe対象システムが使用する文字の構成

[79] 以下のXML及びSGMLの文書仕様を指す(2015年1月時点)。
XMLの文書仕様:『日本国特許庁電子文書交換標準仕様 XML編 第5．41版』の「1．1．5 使用文字種」
SGMLの文書仕様:『日本国特許庁電子文書交換標準仕様 特定書類編 第5．3版』の「6．1 文字セット」

3.4.2.1 使用する文字コードのルール

目的:	使用する文字コードを統一することにより，サブシステム間での認識齟齬を防止するほか，不要な文字コード変換を抑止する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	内部システムで扱う文字コードは，「(1)使用する文字コード」に示す文字コードを採用すること。
特例1:	プログラムプロダクトのログや設定ファイル等，業務処理に直接影響しないものは，「(1)使用する文字コード」に示したルールの適用外とする。
規約2:	内部システムで使用する文字は，「(2)使用可能な文字」に示す文字とすること。
規約3:	内部システムで扱う文字コードの仕様への対応は，「(3)文字コードの仕様への対応」のとおりとすること。

目的:	使用する文字コードを統一することにより，サブシステム間での認識齟齬を防止するほか，不要な文字コード変換を抑止する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	特許庁外字の扱いは，「(4)特許庁外字の扱い」のとおりとすること。

(1) 使用する文字コード

内部システムで使用する文字コードのルールを以下に示す。

- 文字セット／エンコードは，Unicode／UTF-8とする。

(2) 使用可能な文字

Unicodeで規定する文字の中から，内部システムが使用可能な文字を制限する。

特許庁システムの文書仕様で規定される使用可能な文字がJIS第三，第四水準を含むJIS X 0213に拡張されることを想定し，内部システムで使用可能な文字（以降，ToBe使用可能文字と呼ぶ）を以下とする。

- JIS X 0213で規定する文字（以降，ToBe規定内文字と呼ぶ）。
- JIS X 0213で規定外の文字のうち，特許庁外字に該当する文字（以降，ToBe特許庁外字と呼ぶ）。

ToBe対象システムと内部システムにおける規定内文字と特許庁外字の関係を下図に示す。

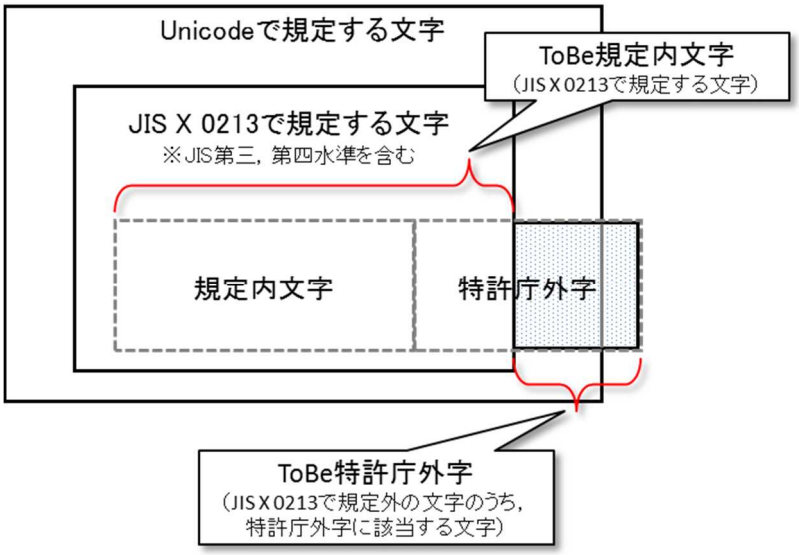


図 3.4-3 ToBe対象システムと内部システムにおける規定内文字と特許庁外字の関係

ToBe規定内文字は，規定内文字を含む。また，ToBe規定内文字は，特許庁外字のうちJIS X 0213で規定している文字も含む。ToBe特許庁外字は，特許庁外字のうちJIS X 0213で規定していない文字を含む。

(3)　文字コードの仕様への対応

JIS第三，第四水準，Unicodeが持つ仕様への対応を以下に示す。

- 「サロゲートペア」は使用する。
- 「結合文字」は使用しない。
- 「異体字セレクタ」[80]は使用しない。

(4)　特許庁外字の扱い

特許庁外字の扱いのルールを以下に示す。

- 各ToBe対象システムで使用している特許庁外字をUnicodeへマッピングし，ToBe特許庁外字を一元化する。

マッピングの結果，特許庁外字は以下のように扱う。

① Unicode及びJIS X 0213に含まれる特許庁外字は，ToBe規定内文字として扱う。

② Unicodeに含まれるが，JIS X 0213に含まれない特許庁外字は，ToBe特許庁外字として扱う。JIS X 0213以外に追加で使用する文字を明確にすること。

③ Unicodeに含まれない特許庁外字は，Unicodeの私用領域にマッピングし，ToBe特許庁外字として扱う。JIS X 0213以外に追加で使用する文字と対応する符号位置を明確にすること。

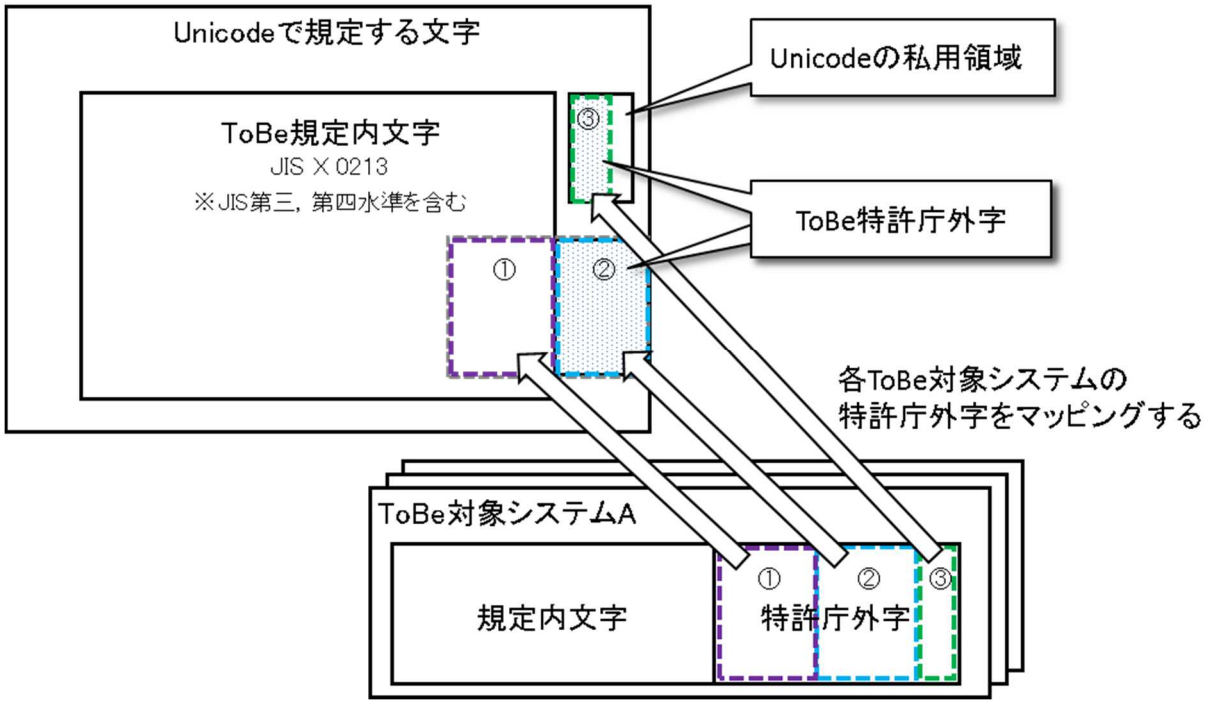


図 3.4-4 特許庁外字のマッピング

[80] Unicodeにおいて異体字のグリフ(字形)を指定するためのセレクタ(選択子)。
異体字は，意味が同じで字形(グリフ)が異なる文字であり，Unicodeの符号位置は親文字と同じである。
文字符号の後に異体字セレクタを置くことで，どの異体字を表すか指定する。
漢字の異体字のコレクションは複数存在し，それぞれで定義する文字が異なり重複も存在するため，
異体字セレクタを使用する場合は適用するコレクションを決定しておく必要がある。また，対応するフォントもそれぞれ異なる。

3.4.2.2 文字コードに関する制御のルール

目的:	使用する文字コードの制御方法を統一することにより，文字コードを制御する機能の拡散を抑制し，変更の影響箇所数を削減する。また，段階的刷新によって新旧システムが混在することを考慮し，刷新により特許庁業務が阻害されないようにする。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	内部システムにおける文字コードに関する制御は，以下に示す(1)，(2)，(4)のとおり扱うこと。
規約2:	特許庁システムの文書仕様における使用可能な文字の変更前後の対応は，(5)のとおりとすること。

(1) 外部システム連携における文字コードの扱い

外部システム連携における文字コードの扱いにおけるルールを以下に示す。

- 外部インタフェースで文字コード変換や外字変換が必要な場合，外部システム連携層で文字コード変換を行うこと。
- 外部システム連携層にて，外部システムから内部システムへ送信された文字種のチェックを行い，ToBe使用可能文字以外の場合は，エラーを返却すること。
- 外部システム連携層にて，内部システムから外部システムへ送信する文字種のチェックを行い，外部システムが対応不可能な文字が含まれている場合は，エラーを返却すること。

(2) 画面における文字コードの扱い

画面における文字コードの扱いにおけるルールを以下に示す。

当該ルールは内部システム（各サブシステムのプレゼンテーションロジックから出力されブラウザ上に配信されたアプリケーション）に適用されるものであり，外部システムからのアクセスについては，上記の「(1) 外部システム連携における文字コードの扱い」にてルールを示している。

- 画面入力又はファイルアップロードで内部システムへ送信された文字種のチェックを行い，ToBe使用可能文字以外の場合は，エラーを返却すること。

(3) 帳票及び出力装置における文字コードの扱い

帳票及び出力装置における文字コードの扱いについては，「3.1.2.5 帳票出力方式」での記載のとおり，本書では扱わないものとする。

(4) 業務アプリケーションにおける文字コードの扱い

業務アプリケーションにおける文字コードの扱いにおけるルールを以下に示す。

- サロゲートペアが含まれても正しく機能する文字列処理をアプリケーションに実装すること。
- 結合文字，異体字セレクタは使用しないものとし，文字種チェックにおいて結合文字，異体字を検出し，エラーを返却すること。

(5)　文書仕様における使用可能な文字の変更前後の対応

特許庁システムの文書仕様で規定される使用可能な文字がJIS第三，第四水準を含むJIS X 0213に拡張されるまでは，規定内文字，特許庁外字のみを扱う必要がある。使用可能な文字の規定が変更される前後の対応のルールを以下に示す。

- 使用可能な文字の規定が変更される前後ともに，下表に示すシステム構成要素で内部システムに送信されるデータの文字種チェックを行うこと。

表 3.4-3 文字種チェックの対象データとチェック箇所

項番	データ	文字種チェックを行うシステム構成要素
1	画面からの入力データ	プレゼンテーションロジック
2	画面からのアップロードされたファイル	業務処理と同じタイミングで文字種チェックを行う必要がある場合（例えば，ファイルのストリームから順次データを取得し，業務処理を行う場合）は業務アプリケーション（ユーザ）。 それ以外は，プレゼンテーションロジックとする。
3	外部システムからの電文又は連携ファイル	ESB又は外部システム互換機能

- 使用可能な文字の規定が変更される前後で，上記の文字種チェックの内容を切り替えること。文字種チェックの内容を下表に示す。

表 3.4-4 文字種チェックの内容

使用可能な文字の規定の変更前／後	文字種チェックの内容
変更前	規定内文字及び特許庁外字に該当する文字以外の場合， エラーを応答する。
変更後	ToBe規定内文字及びToBe特許庁外字に該当する文字以外の場合， エラーを応答する。

3.4.3 開発言語の選定ルール

目的:	業界標準の開発言語を採用することにより，ベンダロックインを排除する。
スコープ:	階層定型化サブシステム及びインタフェース定型化サブシステム
規約1:	「表 3.4-5 システム構成要素毎の開発言語と選定指針」で規約とするシステム構成要素は，同表に示す開発言語を使用すること。
指針1:	「表 3.4-5 システム構成要素毎の開発言語と選定指針」で推奨とするシステム構成要素は，同表の「選定指針」に従い開発言語を採用すること。
推奨1:	「表 3.4-5 システム構成要素毎の開発言語と選定指針」で示す開発言語を推奨する。

(1) 開発言語選定における共通指針

各システム構成要素の開発言語を選定する上での共通指針を以下に示す。

開発言語は，以下の特性を持つ言語を選定すること。

- 開発・保守の容易性
  - 市場のシェアを多く占めており，技術者の確保が容易であること。
  - 開発環境が提供される等，開発生産性が高い開発言語であること。
- 移植の容易性
  - プログラムプロダクトの変更，バージョンアップ等の環境変化が発生しても，対応が容易であること。
- 動作環境に対する対応性
  - 開発したプログラムが動作するプログラムプロダクトがサポートしていること。
- 利用部品に対する対応性
  - 基盤API等のサブシステム間共通機能を呼び出すことができること。
  - コンポーネントが利用するライブラリを呼び出すことができること。

  ライブラリの例： BPMSのワークフローAPI，OCRソフトウェア等

(2) 開発言語の選定

前述の「(1)開発言語選定における共通指針」に従い，開発言語を選定する。下表にシステム構成要素毎の開発言語を示す。

表 3.4-5 システム構成要素毎の開発言語と選定指針

項番	システム構成要素	規約／推奨	開発言語	選定指針
1	ブラウザ	規約	● HTML ● JavaScript ※文書構造はW3C勧告に従うものとし，特定のブラウザ製品でのみ動作する機能は利用しない。	—
2	リッチクライアント	推奨	● .NET Frameworkがサポートする開発言語。	● 「(1)開発言語選定における共通指針」に準ずる。
3	プレゼンテーションロジック	推奨	● Java	● 業務アプリケーション(ユーザ)と同一とする。
4	業務アプリケーション(ユーザ)	推奨	● Java	● 「(1)開発言語選定における共通指針」に準ずる。
5	業務アプリケーション(システム)	推奨	● Java	● 業務アプリケーション(ユーザ)と同一とする。
6	業務アプリケーション(バッチ)	推奨	● Java	● 業務アプリケーション(ユーザ)と同一とする。
		推奨	ジョブ管理エージェントからシェルスクリプトを経由してバッチを起動する場合 ● シェルスクリプト	● 「(1)開発言語選定における共通指針」に準ずる。
7	BPMS	—	—	—

項番	システム構成要素	規約／推奨	開発言語	選定指針
8	BPMS補完機能	推奨	Java	● BPMSがサポートしている開発言語とする。 ● 極力，業務アプリケーション（ユーザ）と同一とする。
9	BRMS	―	―	―
10	ESB	―	―	―
11	外部システム互換機能	推奨	Java	● ESBがサポートしている開発言語 ● 極力，業務アプリケーション（ユーザ）と同一とする。
12	個別データベース ※個別データベースを扱うための言語[81]を示す。	規約	● ANSI／ISOのSQL標準のSQL ● 製品固有の拡張SQL（性能要件を満たす等の目的での使用に制限する）。	―
13	DBアクセス基盤サービス	推奨	Java	● 「(1)開発言語選定における共通指針」に準ずる。
14	共有データベース ※共有データベースを扱うための言語を示す。	規約	● ANSI／ISOのSQL標準のSQL ● 製品固有の拡張SQL（性能要件を満たす等の目的での使用に制限する）。	―

[81] データベースのデータ構造を定義するDDL，データベースのデータを操作するDML，データベースへのアクセス権を制御するDCLの三つの言語のこと。

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール				適合基準	想定記載設計成果物	ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容			記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	
1	13	3.1.1.3 多階層構造の詳細（その１）	規約1	サブシステムは「表 3.1-4 サブシステムの分類とシステム構成要素」に示された「サブシステムの分類」に分類すること。	1	細則なし	以下のいずれかに該当するサブシステムを「階層定型化サブシステム」に分類していること。 (A)産業財産権法に基づく事務処理を行うシステム (B)以下の特徴をすべて持つシステム ・業務的な手続き（受付，審査，登録等）の流れがある。 ・業務的な手続きにより出願に係る業務データの状態が変化する。 ・業務データの状態によって実施されうる手続きが管理される。 以下のすべてに該当するサブシステムを「インタフェース定型化サブシステム」に分類していること。 (C)階層定型化サブシステムの基準に合致しないシステム (D)事務処理業務と共通事項のない特殊な資務を担うシステム													
			規約2	サブシステムの機能は，「表 3.1-3 システム構成要素の種類及び資務」に示された資務を担うシステム構成要素から実現すること。	1	サブシステムを実現する機能は，その資務に応じて「表3.1-3」に示されたいずれかのシステム構成要素に分類すること。	サブシステムの機能とそれを実現するシステム構成要素の対応関係が設計成果物に明記されており，その内容が細則に則っていること。													
			規約3	サブシステムの機能は，「表 3.1-4 サブシステムの分類とシステム構成要素」に示された，サブシステムの分類毎に定められたシステム構成要素群で構成すること。	1	サブシステムは，そのサブシステムの分類に応じて「表 3.1-4」に示されたシステム構成要素群で構成すること。	サブシステムを構成するシステム構成要素に，「表 3.1-4」に示されたシステム構成要素群以外のシステム構成要素が存在しないこと。													
			規約4	システム構成要素間のアクセスは，「3.1.1.3.1.3 (1)多階層構造の概要に基づくアクセスパス」に示すアクセスパスとすること。また，「表 3.1-7 アクセスパスの制限事項」に示すとおりアクセスパスを制限すること。	1	アクセスパスの存在は，「表 3.1-9」，「表 3.1-10」に示されたアクセスパスに限定すること。	システム構成要素間のアクセスパスが設計成果物に明記されており，その内容が細則に則っていること。													
					2	アクセスパスの向きは，「表 3.1-9」，「表 3.1-10」に示された連携元から連携先へのアクセスに限定すること。	システム構成要素間のアクセスパスの向きが設計成果物に明記されており，その内容が細則に則っていること。													
					3	業務アプリケーション（システム）が「ビジネスプロセスのシステムタスクに対応するビジネスロジックを実行するためのサービス」を提供する場合，そのサービスを利用するシステム構成要素は以下に限ること。 ・BPMS ・他サブシステムのプレゼンテーションロジック ※一つのアクティビティのみに相当する処理で，ビジネスプロセスとして意味を成さないため，BPMSを経由するオーバヘッドを考慮し，直接システムタスク相当の処理を呼び出す場合 ・ESB ※外部システムから業務単位のビジネスロジック実行要求を受け付ける場合	業務アプリケーション（システム）の提供するサービスと利用するシステム構成要素が設計成果物に明記されており，その内容が細則に則っていること。													
					4	業務アプリケーション（システム）が「共通リソースデータの取得及び更新のためのサービス」を提供する場合，そのサービスを利用するシステム構成要素は以下に限ること。 ・業務アプリケーション（ユーザ） ・業務アプリケーション（システム） ※No.3のサービスから利用する場合のみ ・業務アプリケーション（バッチ） ・他サブシステムのプレゼンテーションロジック ・ESB ※外部システムから共通リソースデータの操作要求を受け付ける場合	同上													
					5	業務アプリケーション（システム）が「サブシステムが管理する個別データの取得及び更新のためのサービス」を提供する場合，そのサービスを利用するシステム構成要素は以下に限ること。 ・他サブシステムのプレゼンテーションロジック ※更新においては，サブシステムをまたがった業務を同時に扱う必要がある場合のみ（例えば，国際調査と国際予備審査のセット起案）	同上													
					6	業務アプリケーション（システム）が「BPMSサービスの補完処理に必要なデータ取得及び更新のためのサービス」を提供する場合，そのサービスを利用するシステム構成要素は以下に限ること。 ・BPMS補完機能	同上													
			特例1	「3.1.1.3.1.3 (4)アクセスパスの特例」に示すアクセスパスは，下位層のシステム構成要素から上位層のシステム構成要素へのアクセスパスを許容する。	1	「表 3.1-8」のアクセス元からアクセス先へのアクセスパスは，同表の許容されるアクセス条件でアクセスする場合に限ること。	下位層のシステム構成要素から上位層のシステム構成要素へのアクセスパスを設ける条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。													
			特例2	「3.1.1.3.1.3 (5)システム構成要素の配置の特例及び特例となるシステム構成要素のアクセスパス」に示すとおり，特例となるシステム構成要素の配置及びアクセスパスを許容する。	1	業務アプリケーションをUI層に配置し，DBアクセス基盤サービスへ直接アクセスパスを持たせる場合は，以下の条件を満たす場合に限ること。 ・ 当該業務アプリケーションからDBアクセス基盤サービスへのアクセスは参照のみとし，共有データの更新は行わないこと。 ・ この特例となるアクセス先のDBアクセス基盤サービス及び共有データベースは，調査目的のための専用のサーバ環境で稼動することとし，通常の業務処理を行う環境に影響を与えないようにすること。	業務アプリケーションをUI層に配置し，DBアクセス基盤サービスへ直接アクセスパスを持たせる条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。													
			規約5	個別データベースへのアクセスは，「3.1.1.3.1.3 (6) 個別データベースへのアクセスルール」に示すアクセスパスとすること。	1	他サブシステムの個別データベースを直接アクセスすることを禁止し，他サブシステムが提供するサービスを呼び出すことで間接的にアクセスすること 。	個別データベースへのアクセスパスが設計成果物に明記されており，その内容が細則に則っていること。													
			特例1	情報活用系サブシステムへデータ提供する場合は，情報活用系サブシステムからの個別データベースへの直接アクセスを許容する。	1	細則なし	情報活用系サブシステムから個別データベースへ直接アクセスする条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。													
			規約6	各システム構成要素のソフトウェア構成は，「表 3.1-12 ソフトウェア構成一覧」に示す論理ノード及び，ソフトウェア又はデータベースで構成すること。	1	細則なし	各システム構成要素のソフトウェア構成が設計成果物に明記されており，その内容が「表 3.1-12」に則っていること。													

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール					想定記載設計成果物	ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容	適合基準		記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	
2	13	3.1.1.3 多階層構造の詳細（その２）	規約1	サブシステムの処理方式は，「表 3.1-5 サブシステムの処理方式」に示す処理方式とすること。	1	細則なし	サブシステムの処理方式が設計成果物に明記されており，その内容が「表 3.1-5」に則っていること。														
			規約2	サブシステムの処理方式は，「表 3.1-6 処理方式とシステム構成要素」に示された，処理方式毎に定められたシステム構成要素群で構成すること。	1	細則なし	サブシステムの処理方式とそれを実現するシステム構成要素の対応関係が設計成果物に明記されており，その内容が「表 3.1-6」に則っていること。														
3	44	3.1.2.1 APレイヤの定義及びコンポーネントとの関係	規約1	オンライン処理方式のコンポーネント間のアクセスは，「図 3.1-19」，「図 3.1-20」，「図 3.1-22」，「図 3.1-23」，「図 3.1-24」に示すアクセスパスとすること。	1	図においてアクセスパスで連携しているコンポーネント間のみでアクセスを行うこと。	コンポーネント間のアクセスパスが設計成果物に明記されており，その内容が細則に則っていること。														
					2	図に示されたアクセスパスの向きとすること。	コンポーネント間のアクセスパスの向きが設計成果物に明記されており，その内容が細則に則っていること。														
			規約2	オンライン処理方式のコンポーネントは，「表 3.1-19～表 3.1-23」に定義される単位で作成し，同表に定義されている資務を全うするようコンポーネントの機能を実現すること。	1	細則なし	コンポーネントの単位が設計成果物に明記されており，その内容が「表 3.1-19～表 3.1-23」に則っていること。														
			規約3	バッチ処理方式のコンポーネント間のアクセスは，「図 3.1-31」に示すアクセスパスとすること。	1	図においてアクセスパスで連携しているコンポーネント間のみでアクセスを行うこと。	コンポーネント間のアクセスパスが設計成果物に明記されており，その内容が細則に則っていること。														
					2	図に示されたアクセスパスの向きとすること。	コンポーネント間のアクセスパスの向きが設計成果物に明記されており，その内容が細則に則っていること。														
			規約4	バッチ処理方式のコンポーネントは，「表 3.1-29」に定義される単位で作成し，同表に定義されている資務を全うするようコンポーネントの機能を実現すること。	1	細則なし	コンポーネントの単位が設計成果物に明記されており，その内容が「表 3.1-29」に則っていること。														
			規約5	ディレードバッチ処理方式のコンポーネント間のアクセスは，「図 3.1-33」に示すアクセスパスとすること。	1	図においてアクセスパスで連携しているコンポーネント間のみでアクセスを行うこと。	コンポーネント間のアクセスパスが設計成果物に明記されており，その内容が細則に則っていること。														
					2	図に示されたアクセスパスの向きとすること。	コンポーネント間のアクセスパスの向きが設計成果物に明記されており，その内容が細則に則っていること。														
			規約6	ディレードバッチ処理方式のコンポーネントは，「表 3.1-35」に定義される単位で作成し，同表に定義されている資務を全うするようコンポーネントの機能を実現すること。	1	細則なし	コンポーネントの単位が設計成果物に明記されており，その内容が「表 3.1-35」に則っていること。														
4	45	3.1.2.2 オンライン処理方式	指針1	オンライン処理方式のデータ保護方式は，「3.1.2.2.3 データの保護方式」に従って設計すること。	1	一回のリクエストにおける単一データベース内のACID特性を確保する場合は，RDBMSの「システムトランザクション」機能を使用すること。	RDBMSの「システムトランザクション」機能は，一回のリクエストにおける単一データベース内のACID特性を確保するために利用することが設計成果物に明記されていること。														
					2	リクエストをまたがって同一のデータに対して同時に複数のアクセスがある場合や複数データベースをまたがるアクセスがある場合，データ整合性を確保するためにアプリケーションにおいて悲観的排他制御方式又は楽観的排他制御方式を使用すること。	各処理毎に，悲観的排他制御方式又は楽観的排他制御方式のどちらを利用しているかが設計成果物に明記されていること。														
					3	複数データベースをまたがるアクセスがある場合，2フェーズコミット方式は使用しないこと。	複数のデータベースにまたがってアクセスする場合にデータを保護するために2フェーズコミット方式を採用しないことが設計成果物に明記されていること。														
					4	APレイヤ・コンポーネント定義における「ワークフローエンジンアクセス」（BPMSのステータス管理機能によるアクティビティのステータスの更新），「共有DBアクセス（基盤API）」（共有データベースのデータの更新），「個別DBアクセス」（個別データベースのデータの更新）の整合性を確保するための対策を，アプリケーションで取ること。	APレイヤ・コンポーネント定義における「ワークフローエンジンアクセス」（BPMSのステータス管理機能によるアクティビティのステータスの更新），「共有DBアクセス（基盤API）」（共有データベースのデータの更新），「個別DBアクセス」（個別データベースのデータの更新）の整合性を確保するためにアプリケーションが取っている具体的な対策が設計成果物に明記されていること。														
			指針2	オンライン処理方式のエラー処理方式は，「3.1.2.2.4 エラー処理方式」に従って設計すること。	1	「表 3.1-25 エラーの種類」のエラーに対し，「表 3.1-26 エラー発生時の対応」に示す対応を取る設計になっていること。	エラーの種類に対応したエラー処理方式が設計成果物に明記されており，その内容が細則に則っていること。														
					2	システムエラーに関しては，エラー処理を集約的に定義・実行できる仕組み（集約エラーハンドラ）を用いること。	エラー処理を集約的に定義・実行できる仕組みが設計成果物に明記されており，その内容が細則に則っていること。														
					3	モジュールの責任範囲外の値が入力された場合，処理を継続せずに例外を発生させること。	モジュールの責任範囲外の値が入力された場合の対応が設計成果物に明記されており，その内容が細則に則っていること。														
			指針3	オンライン処理方式の業務継続方式は，「3.1.2.2.5 業務継続方式」に従って設計すること。	1	「表 3.1-27 業務継続方式」に記載された業務継続方式に従うこと。	業務継続方式が設計成果物に明記されており，その内容が細則に則っていること。														

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール					想定記載設計成果物	ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容	適合基準		記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価結果 評価日	評価結果 Level-2 (*2)	評価結果 Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価結果 評価日	評価結果 Level-2 (*2)	評価結果 Level-3 (*3)	
5	59	3.1.2.3 バッチ処理方式	指針1	バッチ処理方式のデータ保護方式は，「3.1.2.3.3 データの保護方式」に従って設計すること。	1	一回のバッチにおける単一データベース内のACID特性を確保する場合は，RDBMSの「システムトランザクション」機能を使用すること。	RDBMSの「システムトランザクション」機能は，一回のバッチにおける単一データベース内のACID特性を確保するために利用することが設計成果物に明記されていること。														
					2	バッチをまたがって同一のデータに対して同時に複数のアクセスがある場合や複数データベースをまたがるアクセスがある場合，データ整合性を確保するためにアプリケーションにおいて悲観的排他制御方式又は楽観的排他制御方式を使用すること。	各バッチ処理毎に，悲観的排他制御方式又は楽観的排他制御方式のどちらを利用しているかが設計成果物に明記されていること。												
					3	複数データベースをまたがるアクセスがある場合，2フェーズコミット方式は使用しないこと。	複数のデータベースにまたがってアクセスする場合にデータを保護するために2フェーズコミット方式を採用しないことが設計成果物に明記されていること。														
					4	APレイヤ・コンポーネント定義における「ワークフローエンジンアクセス」（BPMSのステータス管理機能によるアクティビティのステータスの更新），「共有DBアクセス（基盤API）」（共有データベースのデータの更新），「個別DBアクセス」（個別データベースのデータの更新）の整合性を確保するための対策を，アプリケーションで取ること。	APレイヤ・コンポーネント定義における「ワークフローエンジンアクセス」（BPMSのステータス管理機能によるアクティビティのステータスの更新），「共有DBアクセス（基盤API）」（共有データベースのデータの更新），「個別DBアクセス」（個別データベースのデータの更新）の整合性を確保するためにアプリケーションが取っている具体的な対策が設計成果物に明記されていること。												
			指針2	バッチ処理方式のエラー処理方式は，「3.1.2.3.4 エラー処理方式」に従って設計すること。	1	「表 3.1-30 エラーの種類」のエラーに対し，「表 3.1-31 エラー発生時の対応」に示す対応を取る設計になっていること。	エラーの種類に対応したエラー処理方式が設計成果物に明記されており，その内容が細則に則っていること。														
					2	システムエラーに関しては，エラー処理を集約的に定義・実行できる仕組み（集約エラーハンドラ）を用いること。	エラー処理を集約的に定義・実行できる仕組みが設計成果物に明記されており，その内容が細則に則っていること。														
					3	モジュールの責任範囲外の値が入力された場合，処理を継続せずに例外を発生させること。	モジュールの責任範囲外の値が入力された場合の対応が設計成果物に明記されており，その内容が細則に則っていること。														
			指針3	バッチ処理方式の業務継続方式は，「3.1.2.3.5 業務継続方式」に示される業務継続方式のいずれかを採用した設計とすること。	1	「表 3.1-32 業務継続方式」に記載された業務継続方式に従うこと。	業務継続方式が設計成果物に明記されており，その内容が細則に則っていること。														
6	66	3.1.2.4 ディレードバッチ処理方式	指針1	ディレードバッチ処理方式のデータ保護方式は，「3.1.2.4.3 データの保護方式」に従って設計すること。	1	一回のバッチにおける単一データベース内のACID特性を確保する場合は，RDBMSの「システムトランザクション」機能を使用すること。	RDBMSの「システムトランザクション」機能は，一回のバッチにおける単一データベース内のACID特性を確保するために利用することが設計成果物に明記されていること。														
					2	バッチをまたがって同一のデータに対して同時に複数のアクセスがある場合や複数データベースをまたがるアクセスがある場合，データ整合性を確保するためにアプリケーションにおいて悲観的排他制御方式又は楽観的排他制御方式を使用すること。	各バッチ処理毎に，悲観的排他制御方式又は楽観的排他制御方式のどちらを利用しているかが設計成果物に明記されていること。												
					3	複数データベースをまたがるアクセスがある場合，2フェーズコミット方式は使用しないこと。	複数のデータベースにまたがってアクセスする場合にデータを保護するために2フェーズコミット方式を採用しないことが設計成果物に明記されていること。														
					4	APレイヤ・コンポーネント定義における「ワークフローエンジンアクセス」（BPMSのステータス管理機能によるアクティビティのステータスの更新），「共有DBアクセス（基盤API）」（共有データベースのデータの更新），「個別DBアクセス」（個別データベースのデータの更新）の整合性を確保するための対策を，アプリケーションで取ること。	APレイヤ・コンポーネント定義における「ワークフローエンジンアクセス」（BPMSのステータス管理機能によるアクティビティのステータスの更新），「共有DBアクセス（基盤API）」（共有データベースのデータの更新），「個別DBアクセス」（個別データベースのデータの更新）の整合性を確保するためにアプリケーションが取っている具体的な対策が設計成果物に明記されていること。												
			指針2	ディレードバッチ処理方式のエラー処理方式は，「3.1.2.4.4 エラー処理方式」に従って設計すること。	1	「表 3.1-36 エラーの種類」のエラーに対し，「表 3.1-37 エラー発生時の対応」に示す対応を取る設計になっていること。	エラーの種類に対応したエラー処理方式が設計成果物に明記されており，その内容が細則に則っていること。														
					2	システムエラーに関しては，エラー処理を集約的に定義・実行できる仕組み（集約エラーハンドラ）を用いること。	エラー処理を集約的に定義・実行できる仕組みが設計成果物に明記されており，その内容が細則に則っていること。														
					3	モジュールの責任範囲外の値が入力された場合，処理を継続せずに例外を発生させること。	モジュールの責任範囲外の値が入力された場合の対応が設計成果物に明記されており，その内容が細則に則っていること。														
			指針3	ディレードバッチ処理方式の業務継続方式は，「3.1.2.4.5 業務継続方式」に示される業務継続方式のいずれかを採用した設計とすること。	1	「表 3.1-38 業務継続方式」に記載された業務継続方式に従うこと。	業務継続方式が設計成果物に明記されており，その内容が細則に則っていること。														

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール					ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容	適合基準	想定記載設計成果物	記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	
7	74	3.1.2.6 サブシステム間連携処理方式（その１）	指針1	BPMSを介したサブシステム間連携方式は，「3.1.2.6.1.1 (1)連携方式」，「3.1.2.6.1.1 (2)設計・開発時の考慮点」に従って設計すること。	1	BPMSを介してサブシステム間で連携する場合は，以下のいずれかの方式を採用すること。 ・ワークフロー連携（ワークフロー呼出） ・ワークフロー連携（BPMS補完機能呼出） ・サービス連携	BPMSを介してサブシステム間で連携する場合の連携方式が設計成果物に明記されており，その内容が細則に則っていること。													
					2	BPMSのプロセスデータは，連携先サブシステムへ引数で渡すこと。	BPMSのプロセスデータの渡し方が設計成果物に明記されており，その内容が細則に則っていること。													
					3	BPMSのプロセスデータ以外は，共有データベースを用いてデータの受け渡しを行うこと。	BPMSのプロセスデータ以外のデータの渡し方が設計成果物に明記されており，その内容が細則に則っていること。													
					4	共有データベースの特定サブシステム間共有データに一時的に保持される受け渡しデータは，データを受領するサブシステム側で使用後に削除すること。	特定サブシステム間共有データの削除方針が設計成果物に明記されており，その内容が細則に則っていること。													
			指針2	BPMSとジョブ管理システム間の連携方式は，「3.1.2.6.1.2 (1)連携方式」，「3.1.2.6.1.2 (2)設計・開発時の考慮点」に従って設計すること。	BPMSからジョブ管理システムに連携する場合															
					1	「表 3.1-41 BPMSとジョブ管理システム間の連携方式」の項番1に示す方式で連携すること。	BPMSからジョブ管理システムに連携する場合の連携方式が設計成果物に明記されており，その内容が細則に則っていること。													
					2	BPMSのプロセスデータは，連携先サブシステムの業務アプリケーション（システム）へ引数で渡し，BPMSから渡された引数（プロセスデータ）はバッチ処理実行時まで個別データベースに格納しておくこと。	BPMSのプロセスデータの渡し方及びBPMSから渡された引数（プロセスデータ）の格納方法が設計成果物に明記されており，その内容が細則に則っていること。													
					3	BPMSのプロセスデータ以外は，共有データベースを用いてデータの受け渡しを行うこと。	BPMSのプロセスデータ以外のデータの渡し方が設計成果物に明記されており，その内容が細則に則っていること													
					4	バッチ処理実行後に不要となった受け渡しデータ（共有データベースに格納されたプロセスデータ以外のデータ及び個別データベースに格納されたプロセスデータ）は，データを受領するサブシステムが削除すること。	バッチ処理実行後に不要となった受け渡しデータ（共有データベースに格納されたプロセスデータ以外のデータ及び個別データベースに格納されたプロセスデータ）の削除方針が設計成果物に明記されており，その内容が細則に則っていること。													
					5	連携元サブシステムからバッチを起動する契機を与える場合は，以下のいずれかをジョブ管理システムが検知することにより実現すること。 ・連携元サブシステムの要求により連携先サブシステムの個別データベースに格納又は更新されたデータ ・連携元サブシステムの要求により連携先サブシステムに格納された検知用のファイル	連携元サブシステムからバッチを起動する契機を与える場合の方法が設計成果物に明記されており，その内容が細則に則っていること。													
					ジョブ管理システムからBPMSに連携する場合															
					6	「表 3.1-41 BPMSとジョブ管理システム間の連携方式」の項番2に示す方式で連携すること。	ジョブ管理システムからBPMSに連携する場合の連携方式が設計成果物に明記されており，その内容が細則に則っていること。													
					7	データの受け渡しは，No.8の場合を除き共有データベースを用いて行うこと。	データの受け渡し方が設計成果物に明記されており，その内容が細則に則っていること。													
					8	以下に適合する場合は，FTP(S)を使用しファイルにてデータを受け渡すこと。 ・大容量ファイルを効率よく転送したい場合や一括処理のため複数件数のレコードデータを一つのファイルにまとめてやり取りしたい場合等，データベース経由では，効率が極めて悪い場合。	FTP(S)で連携する対象及び左記の条件を満たしていることが設計成果物に明記されていること。													
					9	共有データベースで受け渡したデータは，データを受領するサブシステム側が使用後に削除すること。	共有データベースで受け渡したデータの削除方針が設計成果物に明記されており，その内容が細則に則っていること。													
					10	FTP(S)で送信したファイルは，ファイルを送信するサブシステム側が削除すること。	FTP(S)で送信したファイルの削除方針が設計成果物に明記されており，その内容が細則に則っていること。													
					11	FTP(S)で受信したファイルは，ファイルを受信するサブシステム側が使用後に削除すること。	FTP(S)で受信したファイルの削除方針が設計成果物に明記されており，その内容が細則に則っていること。													
					12	連携元のバッチ処理の終了を契機に，連携先のバッチ処理を開始すること。	連携先のバッチ処理の開始方法が設計成果物に明記されており，その内容が細則に則っていること。													
					13	連携先の業務AP（バッチ）で共有データベースに登録されている複数件のデータを単件単位に分解し，プロセスデータとしてBPMSへ引き渡し，ビジネスプロセスを開始すること。	ビジネスプロセスの開始方法が設計成果物に明記されており，その内容が細則に則っていること。													
			指針3	バッチ処理方式のサブシステム間連携方式は，「3.1.2.6.1.3 (1)連携方式」，「3.1.2.6.1.3 (2)設計・開発時の考慮点」に従って設計すること。	1	他サブシステムのジョブネットとの連携は，時刻起動あるいはFTP(S)等によるファイル到着契機で実行すること。	他サブシステムのジョブネットとの連携方式が設計成果物に明記されており，その内容が細則に則っていること。													
					2	データの受け渡しは，No.3の場合を除き共有データベースを用いて行うこと。	データの受け渡し方が設計成果に明記されており，その内容が細則に則っていること。													
					3	以下に適合する場合は，FTP(S)を使用しファイルにてデータを受け渡すこと。 ・大容量ファイルを効率よく転送したい場合や一括処理のため複数件数のレコードデータを一つのファイルにまとめてやり取りしたい場合等，データベース経由では，効率が極めて悪い場合。	FTP(S)で連携する対象及び左記の条件を満たしていることが設計成果物に明記されていること。													
					4	共有データベースで受け渡したデータは，データを受領するサブシステム側が使用後に削除すること。	共有データベースで受け渡したデータの削除方針が設計成果物に明記されており，その内容が細則に則っていること。													
					5	FTP(S)で送信したファイルは，ファイルを送信するサブシステム側が削除すること。	FTP(S)で送信したファイルの削除方針が設計成果物に明記されており，その内容が細則に則っていること。													
					6	FTP(S)で受信したファイルは，ファイルを受信するサブシステム側が使用後に削除すること。	FTP(S)で受信したファイルの削除方針が設計成果物に明記されており，その内容が細則に則っていること。													

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール					想定記載設計成果物	ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容	適合基準		記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	
			指針4	外部システム連携層を介した外部システム連携方式は，「3.1.2.6.1.4 (1)連携方式」，「3.1.2.6.1.4 (2)設計・開発時の考慮点」，「3.1.2.6.1.4 (4)ESBに関する特記事項」及び「3.1.2.6.1.4 (5)外部システム互換機能に関する特記事項」に従って設計すること。	1	外部システムとの連携は，外部システム連携層を介して連携すること。	外部システム連携層を介した外部システム連携方式が設計成果物に明記されており，その内容が細則に則っていること。														
					2	BPMSのプロセスデータは，連携時に引数で受け渡しを行うこと。	BPMSのプロセスデータの受け渡し方が設計成果物に明記されており，その内容が細則に則っていること。														
					3	BPMSのプロセスデータ以外のデータの受け渡しを行う場合は，No. 4の場合を除き共有データベースを用いること。	BPMSのプロセスデータ以外のデータの受け渡し方が設計成果物に明記されており，その内容が細則に則っていること。														
					4	以下のいずれかに適合する場合は，FTP(S)を使用しファイルにてデータを受け渡すこと。 ・ 大容量ファイルを効率よく転送したい場合や一括処理のため複数件数のレコードデータを一つのファイルにまとめてやり取りしたい場合等，データベース経由では，効率が極めて悪い場合。	FTP(S)で連携する対象及び左記の条件を満たしていることが設計成果物に明記されていること。														
					5	共有データベースで受け渡したデータは，データを受領するサブシステム側が使用後に削除すること。	共有データベースで受け渡したデータの削除方針が設計成果物に明記されており，その内容が細則に則っていること。														
					6	FTP(S)で送信したファイルは，ファイルを送信するサブシステム側が削除すること。	FTP(S)で送信したファイルの削除方針が設計成果物に明記されており，その内容が細則に則っていること。														
					7	FTP(S)で受信したファイルは，ファイルを受信するサブシステム側が使用後に削除すること。	FTP(S)で受信したファイルの削除方針が設計成果物に明記されており，その内容が細則に則っていること。														
					8	サブシステムからESBへの通信は同期で連携すること。	サブシステムからESBへの通信は同期で連携することが設計成果物に明記されていること。														
					9	ESBから外部システムへのアクセスは，内部システムから受信したデータを連携する場合に限ること。	ESBから外部システムへのアクセスするケースが設計成果物に明記されており，その内容が細則に則っていること。														
					10	外部システム刷新後の連携方式がワークフロー連携になる場合，内部システムからESBへはプロセスデータを引数で受け渡すこと。	外部システム刷新後の連携方式がワークフロー連携になるインタフェースが設計成果物に明記されており，当該インタフェースにおいて内部システムからESBへの受け渡しデータがプロセスデータであること。														
					11	ESBからBPMS，BPMS補完機能，業務アプリケーション（システム），DBアクセス基盤サービスへのアクセスは，外部システムから受信したデータを連携する場合に限ること。	ESBからBPMS，BPMS補完機能，業務アプリケーション（システム），DBアクセス基盤サービスへアクセスするケースが設計成果物に明記されており，その内容が細則に則っていること。														
					12	外部システム刷新後の連携方式がワークフロー連携になる場合，ESBからBPMS，BPMS補完機能，業務アプリケーション（システム）へはプロセスデータを引数で受け渡すこと。	外部システム刷新後の連携方式がワークフロー連携になるインタフェースが設計成果物に明記されており，当該インタフェースにおいてESBからBPMS，BPMS補完機能，業務アプリケーション（システム）への受け渡しデータがプロセスデータであること。														
					13	ESBから外部システム互換機能へのアクセスは，内部システムと外部システムとの連携方式のギャップをESBで吸収できない場合かつギャップを吸収する処理を外部システム互換機能で行う場合に限ること。	ESBから外部システム互換機能へアクセスするケースが設計成果物に明記されており，その内容が細則に則っていること。														
					14	外部システム互換機能からDBアクセス基盤サービス又は共有データベースへのアクセスは，ESBで処理できない連携方式のギャップ吸収における，共有データベースへのデータ取得又は更新の場合に限ること。	外部システム互換機能からDBアクセス基盤サービス又は共有データベースへアクセスするケースが設計成果物に明記されており，その内容が細則に則っていること。														
					15	内部システムから外部システムへ渡すデータにプロセスデータ以外のデータがある場合は，外部システム互換機能がDBアクセス基盤サービス又は共有データベースにアクセスして必要なデータを取得すること。	内部システムから外部システムへプロセスデータ以外のデータを渡す方法が設計成果物に明記されており，その内容が細則に則っていること。														
					16	外部システムから内部システムへ渡すデータにプロセスデータ以外のデータがある場合は，外部システム互換機能がDBアクセス基盤サービス又は共有データベースにアクセスして外部システムから取得したデータを格納すること。	外部システムから内部システムへプロセスデータ以外のデータを渡す方法が設計成果物に明記されており，その内容が細則に則っていること。														
8	74	3.1.2.6 サブシステム間連携処理方式（その2）	指針1	サブシステム間連携処理方式のデータ保護方式は，「3.1.2.6.2 データの保護方式」に従って設計すること。	1	一回のリクエスト又はバッチにおける単一データベース内のACID特性を確保する場合は，RDBMSの「システムトランザクション」機能を使用すること。	RDBMSの「システムトランザクション」機能は，一回のリクエスト又はバッチにおける単一データベース内のACID特性を確保するために利用することが設計成果物に明記されていること。														
					2	リクエスト又はバッチをまたがって同一のデータに対して同時に複数のアクセスがある場合や複数データベースをまたがるアクセスがある場合，データ整合性を確保するためにアプリケーションにおいて悲観的排他制御方式又は楽観的排他制御方式を使用すること。	各処理毎に，悲観的排他制御方式又は楽観的排他制御方式のどちらを利用しているかが設計成果物に明記されていること。														
					3	複数データベースをまたがるアクセスがある場合，2フェーズコミット方式は使用しないこと。	複数のデータベースにまたがってアクセスする場合にデータを保護するために2フェーズコミット方式を採用しないことが設計成果物に明記されていること。														
					4	APレイヤ・コンポーネント定義における「ワークフローエンジンアクセス」（BPMSのステータス管理機能によるアクティビティのステータスの更新），「共有DBアクセス（基盤API）」（共有データベースのデータの更新），「個別DBアクセス」（個別データベースのデータの更新）の整合性を確保するための対策を，アプリケーションで取ること。	APレイヤ・コンポーネント定義における「ワークフローエンジンアクセス」（BPMSのステータス管理機能によるアクティビティのステータスの更新），「共有DBアクセス（基盤API）」（共有データベースのデータの更新），「個別DBアクセス」（個別データベースのデータの更新）の整合性を確保するためにアプリケーションが取っている具体的な対策が設計成果物に明記されていること。														
			指針2	サブシステム間連携処理方式の業務継続方式は，「3.1.2.6.3 業務継続方式」に従って設計すること。	1	「表 3.1-27 業務継続方式」に記載された業務継続方式に従うこと。	業務継続方式が設計成果物に明記されており，その内容が細則に則っていること。														
					2	「表 3.1-32 業務継続方式」に記載された業務継続方式に従うこと。	業務継続方式が設計成果物に明記されており，その内容が細則に則っていること。														

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール					想定記載設計成果物	ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容	適合基準		記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	
9	94	3.2.1.1.1 サービス化の指針	規約1	「(4)サービス化の対象」に示すシステム構成要素の提供機能は，サービスインタフェースを設けるよう設計すること。	1	「表 3.2-2」のサービス化対象のシステム構成要素は，同表の提供機能を提供する目的でサービスインタフェースを設けること。	サービス化対象のシステム構成要素が設計成果物に明記されており，その内容が細則に則っていること。														
			特例1	「(3)サービス化対象外の指針」に該当するものについては，サービス化の対象外とすることを許容する。	1	サービス化の対象外とする場合は，以下のいずれかの条件を満たす場合に限ること。 ・ 「(1)サービスの条件」で示す条件を実現できない場合 ・ リアルタイム性が求められサービス化により要求性能が満たせない場合 ・ サービスにアクセスするオーバヘッドが処理時間の多くを占めることにより要求性能が満たせない場合 ・ サービス化により信頼性要件を満たすための実現コストが許容できない場合 ・ 複数のサービスにまたがるトランザクション処理や復旧処理が必要となったり，処理のオーバヘッドによって要求性能が満たせない場合 ・ 複数の分かれた操作にまたがってACID特性が求められるような場合	サービス化対象外とする条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。												
			規約2	サービスインタフェースは，「(5)サービスインタフェースの提供」のとおり提供すること。	1	セッションを使用しないサービスインタフェースを提供すること。	サービスインタフェースの提供形態が設計成果物に明記されており，その内容が細則に則っていること。														
10	96	3.2.1.1.2 内部インタフェースのプロトコル	規約1	「表 3.2-3 プロトコル及びデータフォーマット（階層定型化サブシステム）」及び「表 3.2-4 プロトコル及びデータフォーマット（インタフェース定型化サブシステム）」に示すシステム構成要素は，当該表に示すプロトコル及びデータフォーマットを利用するインタフェースを設けること。	1	細則なし	システム構成要素毎の内部インタフェースのプロトコル及びデータフォーマットが設計成果物に明記されており，その内容が「表 3.2-3」及び「表 3.2-4」に則っていること。														
			特例1	「表 3.2-3 プロトコル及びデータフォーマット（階層定型化サブシステム）」に示す特例を許容する。	1	細則なし	「表 3.2-3」の「（※特例）」と記しているプロトコル及びデータフォーマットの使用条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。														
11	101	3.3.1.1 クライアント構成	規約1	クライアント構成は，「ブラウザを用いた構成」を採用すること。	1	細則なし	ブラウザを用いた構成とする旨が設計成果物に明記されていること。														
			特例1	「(2)リッチクライアント構成の選定ルール」に該当する場合に限り，「リッチクライアントを用いた構成」の採用を許容する。	1	以下のいずれかに該当する場合に限り，リッチクライアントを用いた構成とすること。 ・ブラウザで実現できない高い入力効率が求められる場合 ・ブラウザで実現できない表示が求められる場合 ・OCRのソフトウェア等，ブラウザで扱うことのできないソフトウェアを利用する場合 ・「ブラウザを用いた構成」では，特定のブラウザに依存した機能を利用しないと実現できない場合	リッチクライアントを用いた構成とする条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。														
12	103	3.3.1.2 画面とリクエストの作成単位に関するルール	指針1	ユーザタスクに対応する画面とリクエストは，ユーザタスク毎に閉じた単位で作成すること。	1	細則なし	ユーザタスクに対応する画面とリクエストの単位が設計成果物に明記されており，その内容がルールに則っていること。														
			特例1	「(2)画面とリクエストの作成単位に関する特例」に示した業務に限り，複数のユーザタスクに対し同一のリクエストと画面のまとまりを割り当ててもよい。	1	複数のユーザタスクに対し同一のリクエストと画面のまとまりを割り当てる場合は，以下の業務処理に限ること。 ・ 複数のユーザタスクを同時に進める業務処理 ・ 複数の業務データを横断的に参照する業務処理	複数のユーザタスクに対し同一のリクエストと画面のまとまりを割り当てる条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。														
13	105	3.3.1.3 画面の定義に関するルール	規約1	画面の単位は，「(1)画面の単位」に従って設計すること。	1	以下の場合を除き，アクティビティ（ユーザタスク）をまたがらないよう画面を分けること。 ・ダイアログのようにアクティビティが異なっても共通的に利用される画面で，画面構成等の将来的な変更内容がアクティビティ毎に異ならないことが想定される場合	画面の単位が設計成果物に明記されており，その内容が細則に則っていること。														
					2	機能をまたがらないよう画面を分けること。	画面の単位が設計成果物に明記されており，その内容が細則に則っていること。														
			規約2	画面間の遷移は，「(2)画面間の遷移」に従って設計すること。	1	他のアクティビティに関する画面群に直接遷移しないこと。	アクティビティをまたく画面遷移が設計されていないこと。														
14	107	3.3.1.4 ブラウザからの要求を受けるアプリケーションの粒度	指針1	ブラウザからの要求を受けるアプリケーション（プレゼンテーションロジック及び業務アプリケーション（ユーザ）のまとまり）の粒度は，「(1)ブラウザからの要求を受けるアプリケーションの粒度のルール」に従って設計すること。	1	複数のサブシステムをまたがらないようにアプリケーション（プレゼンテーションロジック及び業務アプリケーション（ユーザ）のまとまり）を構成すること。	各アプリケーション（プレゼンテーションロジック及び業務アプリケーション（ユーザ）のまとまり）に対し，複数のサブシステムが対応付いていないこと。														
					2	サブシステム内において，アプリケーション（プレゼンテーションロジック及び業務アプリケーション（ユーザ）のまとまり）の管理として，適切な業務体系や管理単位の元に分割管理されていること。	アプリケーション（プレゼンテーションロジック及び業務AP（ユーザ）のまとまり）の粒度が設計成果物に明記されており，その内容が細則に則っていること。														
					3	サブシステム内において，バージョンアップによるアプリケーション（プレゼンテーションロジック及び業務アプリケーション（ユーザ）のまとまり）の起動・停止等，運用的な観点で分割管理されていること。	アプリケーション（プレゼンテーションロジック及び業務AP（ユーザ）のまとまり）の粒度が設計成果物に明記されており，その内容が細則に則っていること。														
					4	サブシステム内において，求められる信頼性や性能の度合いが大きく異なる機能は別のアプリケーション（プレゼンテーションロジック及び業務アプリケーション（ユーザ）のまとまり）とすること。	アプリケーション（プレゼンテーションロジック及び業務AP（ユーザ）のまとまり）の粒度が設計成果物に明記されており，その内容が細則に則っていること。														
15	111	3.3.2.1.1 ビジネスプロセス管理として利用する技術	規約1	ビジネスプロセスの実行はBPMSで行うこと。	1	細則なし	BPMSを使用する旨が設計成果物に明記されていること。														

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール					想定記載設計成果物	ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容	適合基準		記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	
16	112	3.3.2.1.2 BPMSの適用範囲に関する設計指針	指針1	BPMSの可視性が確保できないケースでのアプリケーションの資務及び資務を担うシステム構成要素は，「表 3.3-7 BPMSで可視性が確保できないケースでのアプリケーションの資務」のとおりとすること。	1	細則なし	BPMSの可視性が確保できないケースでのアプリケーションの資務及び資務を担うシステム構成要素が設計成果物に明記されており，その内容が「表 3.3-7」に則っていること。															
			指針2	待機制御におけるBPMSとアプリケーションの資務は，「表 3.3-8 待機制御におけるBPMSとアプリケーションの資務の切り分け」に従うこと。	1	細則なし	待機制御におけるBPMSとアプリケーションの資務が設計成果物に明記されており，その内容が「表 3.3-8」に則っていること。															
			指針3	各BPMSの呼び出し元における事前の業務チェックを行う箇所は「表 3.3-9 BPMSの呼び出し元と事前チェック箇所」に従うこと。	1	細則なし	各BPMSの呼び出し元における事前の業務チェックを行う箇所が設計成果物に明記されており，その内容が「表 3.3-9」に則っていること。															
17	128	3.3.2.1.3 プロセスデータとして保持する情報	指針1	ビジネスプロセスのプロセスデータは，BPMSではキー情報だけを保持するように設計すること。	1	細則なし	プロセスデータがキー情報であることが設計成果物に明記されていること。														
18	129	3.3.2.1.4 ビジネスプロセスの状態管理の設計指針	指針1	ビジネスプロセスのステータスの管理は，「(1)状態管理のルール」に従って行うこと。	1	ビジネスプロセスのステータスはBPMSのステータス管理機能を利用して管理すること。	ビジネスプロセスのステータスの管理方法が設計成果物に明記されており，その内容が細則に則っていること。														
			特例1	ビジネスプロセスの状態を一覧取得する際，性能面に懸念がある場合には，「(2)ビジネスプロセスの状態の保持方法」に示す保持方法を許容する。	1	ビジネスプロセスの状態の一部をデータベースにも重複して保持する場合は，以下の業務処理に限ること。 ・BPMS内部で保持しているビジネスプロセスの状態を一覧取得する際，性能面に懸念がある業務処理	ビジネスプロセスの状態の一部をデータベースにも重複して保持する条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。														
19	130	3.3.2.2.1 BPMS補完機能で実現する機能	規約1	BPMS補完機能で実現する機能はBPMS単体で実現できない機能に限り，BPMSと連携し補完する処理を実装すること。	1	細則なし	BPMS補完機能の機能がBPMSのみでは実現できないものであることが設計成果物に明記されていること。														
			規約2	BPMS補完機能には業務アプリケーション層の資務となる業務ロジックは持たせず，ワークフロー層の資務として，起動するビジネスプロセス又は連携するビジネスプロセスインスタンスの特定やイベント通知の要否判定といった制御を行うこと。	1	細則なし	BPMS補完機能の機能が設計成果物に明記されており，その内容がルールに則っていること。														
20	133	3.3.2.2.3 BPMS補完機能の配置位置	規約1	BPMS補完機能の配置は，補完対象のBPMSサービスを提供するサブシステムとすること。	1	細則なし	BPMS補完機能の配置位置が設計成果物に明記されており，その内容がルールに則っていること。														
21	136	3.3.3.1 業務アプリケーション（システム）のサービスの粒度	指針1	サービスの粒度は，「(1)サービスの粒度のルール」に従って設計すること。	1	複数のサブシステムをまたがらないようにサービスを構成すること。	サービスの粒度が設計成果物に明記されており，その内容が細則に則っていること。														
					2	サブシステム内において，サービスの管理として，適切な業務体系や管理単位の元に分割管理されていること。	サービスの粒度が設計成果物に明記されており，その内容が細則に則っていること。														
					3	サブシステム内において，バージョンアップによるサービスの起動・停止等，運用的な観点で分割管理されていること。	サービスの粒度が設計成果物に明記されており，その内容が細則に則っていること。														
					4	サブシステム内において，求められる信頼性や性能の度合いが大きく異なる機能は別のサービスとすること。	サービスの粒度が設計成果物に明記されており，その内容が細則に則っていること。														
22	138	3.3.3.2 業務アプリケーション（システム）のサービスインタフェースの粒度	指針1	サービスインタフェースの粒度は，「(1)サービスインタフェースの粒度のルール」に従って設計すること。	1	ビジネスプロセスのシステムタスクに対応してビジネスロジックを実行する業務アプリケーション（システム）のサービスインタフェースの粒度は，業務として意味のある最小の単位とすること。	ビジネスプロセスのシステムタスクに対応してビジネスロジックを実行する業務アプリケーション（システム）のサービスインタフェースの粒度が設計成果物に明記されており，その内容が細則に則っていること。														
					2	業務単位のサービスインタフェースを設計する際は，粒度のルールに従い，必要に応じて，サービスインタフェースを統合又は分割すること。	ビジネスプロセスのシステムタスクに対応してビジネスロジックを実行する業務アプリケーション（システム）のサービスインタフェースの粒度が設計成果物に明記されており，その内容が細則に則っていること。														
					3	業務アプリケーション（システム）に呼び出し元に依存した処理を実装しないこと。	ビジネスプロセスのシステムタスクに対応してビジネスロジックを実行する業務アプリケーション（システム）のサービスインタフェースの粒度が設計成果物に明記されており，その内容が細則に則っていること。														
23	140	3.3.3.3.1 サブシステム間共通機能の提供形態	規約1	サブシステム間共通機能の提供は，「(2)サブシステム間共通機能提供時の留意事項」に示す事項に留意し，提供すること。	1	共有コンポーネントは，自サブシステム内の独自コンポーネントに依存しないこと。	サブシステム間共通機能の提供形態が設計成果物に明記されており，その内容が細則に則っていること。														
24	141	3.3.3.4.1 個別データベースの構成	規約1	個別データベースに配置するデータは，「(1)個別データベースに配置するデータのルール」に従うこと。	1	個別データベースに配置する対象データは共通リソースデータ，個別リソースデータ及び個別業務イベントデータとすること。	個別データベースに配置する対象データが共通リソースデータ，個別リソースデータ又は個別業務イベントデータであることが設計成果物に明記されていること。														
25	142	3.3.3.4.2 個別データベースへのアクセスルール	指針1	情報活用系サブシステムへのデータ提供は，「(1)情報活用系サブシステムへのデータ提供のルール」に準拠して設計すること。	1	個別データベースはETL製品が持つ汎用データベースアクセスインタフェース（JDBC，ODBC）に対応したライブラリを提供すること。	個別データベースが提供するライブラリが設計成果物に明記されており，その内容が細則に則っていること。														
26	143	3.3.3.5 アプリケーション基盤機能	規約1	ToBeアーキテクチャにおいては，「(1)　②共通部品」に示される機能を有するアプリケーション基盤（以下，「AP基盤」と称す）を構築し，これを利用すること。	1	細則なし	AP基盤機能を構築し，それを利用する旨が設計成果物に明記されていること。														
27	145	3.3.4.1.1 外部システム連携層に利用する技術	指針1	外部システム連携層は，プログラムプロダクトを有効に活用し設計すること。	1	細則なし	アプリケーションに不適切な作り込みが発生しないようプログラムプロダクトを利用している旨が，設計成果物に明記されており，その内容からプログラムプロダクトを有効活用していることが読み取れること。														
			推奨1	外部システム連携層に利用するプログラムプロダクトは，ESBを推奨する。	1	細則なし	外部システム連携層に利用するプログラムプロダクトに推奨されているものを利用していることが設計成果物に明記されていること。														

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール					想定記載設計成果物	ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容	適合基準		記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価日	Level-2 (*2)	Level-3 (*3)	
28	146	3.3.4.1.2 外部システム連携層に持たせる機能（その1）	規約1	外部システム連携層の機能は，「表 3.3-19 外部システム連携層が持つ機能の利用可否」の利用可否に従い機能を提供すること。	1	細則なし	外部システム連携層の機能が設計成果物に明記されており，その内容がルールに則っていること。														
			特例1	「表 3.3-19 外部システム連携層が持つ機能の利用可否」に示す特例を許容する。	1	「表 3.3-19 外部システム連携層が持つ機能の利用可否」における利用不可の機能を特例で利用可にする際は，「※特例」の記載の場合に限ること。	「表 3.3-19」における利用不可の機能を特例で利用可にする条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。														
29	146	3.3.4.1.2 外部システム連携層に持たせる機能（その2）	指針1	外部システム連携層で吸収するギャップは，「(2)外部システム連携層が吸収するギャップ」に記載したものを対象とするように設計すること。	1	外部システム連携層は，「表 3.3-20 外部システム連携層が吸収するギャップ」に示す以下のギャップを吸収すること。 ・プロトコル ・文字コード ・データフォーマット ・連携データ ・連携タイミング ・通信データ量	外部システム連携層が吸収するギャップが設計成果物に明記されており，その内容が細則に則っていること。														
			指針2	外部システム連携層のプログラムプロダクトにESBを利用する場合，ESBと外部システム互換機能の責務の分担は，「(3)ESBと外部システム互換機能の責務の分担」に従って設計すること。	1	外部システム連携層のプログラムプロダクトにESBを利用する場合，ESB製品が保有する機能で以下のギャップを吸収すること。 ・プロトコル ・文字コード ・データフォーマット ・連携データ	外部システム連携層が吸収するギャップが設計成果物に明記されており，その内容が細則に則っていること。														
					2	ESB製品が保有する機能でNo.1に示すギャップが吸収できない場合，外部システム互換機能が吸収すること。	外部システム互換機能が吸収するギャップが設計成果物に明記されており，その内容が細則に則っていること。														
					3	外部システム互換機能で以下のギャップを吸収すること。 ・連携タイミング ・通信データ量	外部システム互換機能が吸収するギャップが設計成果物に明記されており，その内容が細則に則っていること。														
30	151	3.3.5.1 共有データベースに配置するデータ及びアクセスルール	規約1	共有データベースに配置するデータは，「(1)共有データベースに配置するデータのルール」に従うこと。	1	共有データベースに配置するデータは事件・書類データ，特定サブシステム間共有データとすること。	共有データベースに配置する対象データが事件・書類データ，特定サブシステム間共有データであることが設計成果物に明記されていること。														
			規約2	共有データベースへのアクセスは，「(2)共有データベースへのアクセスルール」に準拠すること。	1	以下のシステム構成要素からのみアクセスすること。 ・ 業務アプリケーション（ユーザ） ・ 業務アプリケーション（システム） ・ 業務アプリケーション（バッチ） ・ DBアクセス基盤サービス ・ 外部システム互換機能	共有データベースにアクセスするシステム構成要素が設計成果物に明記されており，その内容が細則に則っていること。														
					2	以下の使用条件のいずれかに合致する場合，基盤API（サービス版）を使用すること。 ・ データを取得する場合 ・ 一トランザクションに閉じた更新を行う場合	基盤API（サービス版）を使用する業務処理が設計成果物に明記されており，その内容が使用条件を満たしていること。														
					3	以下の使用条件のいずれかに合致する場合，基盤API（データベース版）を使用すること。 ・ 基盤API（サービス版）で用意する単位をまたぐ（複数を一括でまとめる）単位で，更新を含む業務上のトランザクションを必要とする場合 ・ 基盤API（サービス版）を使用することにより，要求性能が満たせない場合	基盤API（データベース版）を使用する業務処理が設計成果物に明記されており，その内容が使用条件を満たしていること。														
					4	特定サブシステム間共有データは，データの授受を行う，送信元サブシステムと送信先サブシステムのみアクセス可能とすること。	・特定サブシステム間共有データの送信元と送信先が設計成果物に明記されていること。 ・特定サブシステム間共有データが送信元と送信先のみアクセス可能である旨が設計成果物に明記されていること。														
			指針1	情報活用系サブシステムへのデータ提供は，「(3)情報活用系サブシステムへのデータ提供のルール」に準拠して設計すること。	1	共有データベースはETL製品が持つ汎用データベースアクセスインタフェース（JDBC，ODBC）に対応したライブラリを提供すること。	共有データベースが提供するライブラリが設計成果物に明記されており，その内容が細則に則っていること。														
31	156	3.3.6.1.1 ビジネスルールをアプリケーションから切り離すための仕組みとして利用する技術（その1）	規約1	ビジネスルールを業務アプリケーション又はBPMSから切り離すための仕組みは，BRMSを用いること。	1	細則なし	BRMSを使用する旨が設計成果物に明記されていること。														
32	156	3.3.6.1.1 ビジネスルールをアプリケーションから切り離すための仕組みとして利用する技術（その2）	規約1	BRMSで使用するデータは，「(3)BRMSで使用するデータ取得に関するルール」に従うこと。	1	BRMSから直接データベースにアクセスしデータを取得しないこと。	BRMSで使用するデータの取得方法が設計成果物に明記されており，その内容が細則に則っていること。														
					2	データベースからのデータ取得は，BRMSの呼び出し元である業務アプリケーションで行い，BRMSへ渡すこと。	BRMSで使用するデータの取得方法が設計成果物に明記されており，その内容が細則に則っていること。														
			指針1	BRMSの適用は，「(2)BRMSを用いたビジネスルールの記述対象」に従って設計すること。	1	BRMSを用いたビジネスルールの記述対象は，「推論」，「計算」のビジネスルールとすること。	BRMSを用いたビジネスルールの記述対象が設計成果物に明記されており，その内容が細則に則っていること。														
					2	BRMSを用いたビジネスルールの記述対象は，以下の二つの特性を併せ持つビジネスルールとすること。 ・ 複数の「定型的な判定条件群と，結果」の組み合わせが多数現れるビジネスルール ・ 変更が発生しやすいビジネスルール又は複数箇所から利用されるビジネスルール	BRMSを用いたビジネスルールの記述対象が設計成果物に明記されており，その内容が細則に則っていること。														
			特例1	ビジネスルールは，「(2)BRMSを用いたビジネスルールの記述対象」に示す特例を許容する。	1	指針1を満たすビジネスルールでもBRMSを用いずにビジネスロジックに記述する際は，以下を満たす場合に限ること。 ・無理にビジネスルールとデータベースアクセス等を分けて管理するとかえって保守性が悪くなる場合	ビジネスルールをBRMSを用いずにビジネスロジックに記述する条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。														
					2	BRMS製品の持つルールのフロー制御機能を使用する際は，以下の2点を満たす場合に限ること。 ・ルールの可視性が低下しない。 ・各ルールの再利用性が妨げられない。	BRMS製品のルールフロー制御機能を使用する条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。														
33	158	3.3.6.1.2 BRMSの呼び出し方式	規約1	BRMSの呼び出し方式は，「サービス方式」とすること。	1	細則なし	BRMSの呼び出し方式が設計成果物に明記されており，その内容がルールに則っていること。														

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール					想定記載設計成果物	ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容	適合基準		記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価結果 評価日	評価結果 Level-2 (*2)	評価結果 Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価結果 評価日	評価結果 Level-2 (*2)	評価結果 Level-3 (*3)	
34	159	3.4.1.1 XMLとSGMLのデータの扱いのルール	指針1	XMLやSGMLの書類形式を意識した業務処理は，「図 3.4-1 書類形式を意識する層の範囲」に示す範囲に配置するよう設計すること。	1	XMLやSGMLの書類形式を意識した業務処理（書類形式の判定処理，スキーマチェック，パース処理等）は，多階層構造における以下の層に配置し，上位層に対して書類形式を隠蔽すること。 ・ 業務アプリケーション層 ・ 基盤機能層 ・ データベース層 ・ 外部システム連携層（外部システムと内部システムとの連携時に扱う書類形式が異なる場合のみ） ・ ビジネスルール管理層	XMLやSGMLの書類形式を意識した業務処理（書類形式の判定処理，スキーマチェック，パース処理等）の配置が設計成果物に明記されており，その内容が細則に則っていること。													
			特例1	書類形式を保持したまま，画面での処理を行う業務要件がある場合には，UI層，プレゼンテーション層での処理を許容する。	1	細則なし	UI層，プレゼンテーション層にXMLやSGMLの書類形式を意識した業務処理を配置する条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。													
35	161	3.4.1.2 処分が決定した事件データの扱いのルール	指針1	処分が決定した事件データは，「表 3.4-1 処分が決定した事件データの扱い方針」に示す方針に従い設計すること。	1	処分が決定した事件データは，データの種類毎に保存期間を設けること。	データの種類とその保存期間が設計成果物に明記されていること。													
					2	処分が決定した事件データは，保存期間を超過したデータをデータベースから削除する仕組みを用意すること。	処分が決定した事件データの保存期間超過後の扱いが設計成果物に明記されており，その内容が細則に則っていること。													
					3	処分が決定した事件データは，事件の権利消滅後の保存期間内において，アプリケーションからのデータアクセス方法を権利消滅前のアクセス方法と変えないこと。	処分が決定した事件データの保存期間内の扱いが設計成果物に明記されており，その内容が細則に則っていること。													
36	163	3.4.2.1 使用する文字コードのルール（その１）	規約1	内部システムで扱う文字コードは，「(1)使用する文字コード」に示す文字コードを採用すること。	1	内部システムで扱う文字コードは，以下とすること。 文字セット：Unicode エンコード：UTF-8	内部システムで扱う文字コードが設計成果物に明記されており，その内容が細則に則っていること。													
			特例1	プログラムプロダクトのログや設定ファイル等，業務処理に直接影響しないものは，「(1)使用する文字コード」に示したルールの適用外とする。	1	細則なし	「（1）使用する文字コード」以外の文字コードを使用する条件が設計成果物に明記されており，その内容が特例の条件を満たしていること。													
			規約2	内部システムで使用する文字は，「(2)使用可能な文字」に示す文字とすること。	1	内部システムで使用可能な文字は，以下とすること。 ・ JIS X 0213で規定する文字（ToBe規定内文字） ・ JIS X 0213で規定外の文字のうち，特許庁外字に該当する文字（ToBe特許庁外字）	内部システムで使用する文字が設計成果物に明記されており，その内容が細則に則っていること。													
			規約3	内部システムで扱う文字コードの仕様への対応は，「(3)文字コードの仕様への対応」のとおりとすること。	1	サロゲートペアは使用すること。	内部システムで扱う文字コードの仕様への対応が設計成果物に明記されており，その内容が細則に則っていること。													
					2	結合文字は使用しないこと。	内部システムで扱う文字コードの仕様への対応が設計成果物に明記されており，その内容が細則に則っていること。													
					3	異体字セレクタは使用しないこと。	内部システムで扱う文字コードの仕様への対応が設計成果物に明記されており，その内容が細則に則っていること。													
37	163	3.4.2.1 使用する文字コードのルール（その２）	規約1	特許庁外字の扱いは，「(4)特許庁外字の扱い」のとおりとすること。	1	各ToBe対象システムで使用している特許庁外字をUnicodeへマッピングすること。	特許庁外字の扱いが設計成果物に明記されており，その内容が細則に則っていること。													
					2	Unicode及びJIS X 0213に含まれる特許庁外字は，ToBe規定内文字として扱うこと。	特許庁外字の扱いが設計成果物に明記されており，その内容が細則に則っていること。													
					3	Unicodeに含まれるが，JIS X 0213に含まれない特許庁外字は，ToBe特許庁外字として扱うこと。	特許庁外字の扱いが設計成果物に明記されており，その内容が細則に則っていること。													
					4	Unicodeに含まれるが，JIS X 0213に含まれない特許庁外字について，対象の文字を明確にすること。	Unicodeに含まれるが，JIS X 0213に含まれない特許庁外字について，対象の文字が設計成果物に明記されていること。													
					5	Unicodeに含まれない特許庁外字は，Unicodeの私用領域にマッピングし，ToBe特許庁外字として扱うこと。	特許庁外字の扱いが設計成果物に明記されており，その内容が細則に則っていること。													
					6	Unicodeに含まれない特許庁外字について，対象の文字と対応するUnicode上の符号位置を明確にすること。	Unicodeに含まれない特許庁外字について，対象の文字と対応するUnicode上の符号位置が設計成果物に明記されていること。													

別紙1　特許庁アーキテクチャ標準仕様書　第1.1版　　適合確認チェックリスト（本冊）

項番	頁	ルール名	ルール					想定記載設計成果物	ＡＰベンダ記入欄						ＪＰＯ記入欄						備考
			強制力	内容	細則 No.	細則 内容	適合基準		記載箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価結果 評価日	評価結果 Level-2 (*2)	評価結果 Level-3 (*3)	確認箇所 ※設計成果物名と章項節を記載	コメント ※評価が「適合」以外の場合は必ず記載	評価結果 Level-1 (*1)	評価結果 評価日	評価結果 Level-2 (*2)	評価結果 Level-3 (*3)	
38	165	3.4.2.2 文字コードに関する制御のルール	規約1	内部システムにおける文字コードに関する制御は，以下に示す(1)，(2)，(4)のとおり扱うこと。	1	外部インタフェースで文字コード変換や外字変換が必要な場合，外部システム連携層で文字コード変換を行うこと。	外部インタフェースで文字コード変換や外字変換が必要な場合に，どこで文字コード変換を行うかが設計成果物に明記されており，その内容が細則に則っていること。														
					2	外部システム連携層にて，外部システムから内部システムへ送信された文字種のチェックを行い，ToBe使用可能文字以外の場合は，エラーを返却すること。	以下の事項全てが設計成果物に明記されていること。 ①　外部システムから内部システムへ送信された文字種のチェックを行うこと。 ②　①の文字種チェックは外部システム連携層で行うこと。 ③　①の文字種チェックでToBe使用可能文字以外の場合にエラーを返却すること。												
					3	外部システム連携層にて，内部システムから外部システムへ送信する文字種のチェックを行い，外部システムが対応不可能な文字が含まれている場合は，エラーを返却すること。	以下の事項全てが設計成果物に明記されていること。 ①　内部システムから外部システムへ送信された文字種のチェックを行うこと。 ②　①の文字種チェックは外部システム連携層で行うこと。 ③　①の文字種チェックでToBe使用可能文字以外の場合にエラーを返却すること。												
					4	画面入力又はファイルアップロードで内部システムへ送信された文字種のチェックを行い，ToBe使用可能文字以外の場合は，エラーを返却すること。	以下の事項全てが設計成果物に明記されていること。 ①　画面入力又はファイルアップロードで送信された文字種のチェックを行うこと。 ②　①の文字種チェックでToBe使用可能文字以外の場合は，エラーを返却すること。												
					5	サロゲートペアが含まれても正しく機能する文字列処理をアプリケーションに実装すること。	サロゲートペアを考慮した文字列処理が設計成果物に明記されていること。														
					6	結合文字，異体字セレクタは使用しないものとし，文字種チェックにおいて検出し，エラーを返却すること。	以下の事項全てが設計成果物に明記されていること。 ①　結合文字，異体字セレクタを使用しないこと。 ②　結合文字，異体字セレクタの文字種チェックを行うこと。 ③　②の文字種チェックで結合文字，異体字セレクタを検出した場合はエラーを返却すること。												
			規約2	特許庁システムの文書仕様における使用可能な文字の変更前後の対応は，(5)のとおりとすること。	1	文字種チェックの対象データとチェック箇所は「表 3.4-3」に従うこと。	表の各データに対し，文字種チェックを行うプログラムが，表の文字種チェックを行うシステム構成要素に属していること。														
					2	使用可能な文字の規定が変更される前後で，「表 3.4-4」に従い，文字種チェックの内容を切り替えること。	・使用可能文字の規定の変更前 　規定内文字及び特許庁外字に該当する文字の文字種チェック処理が設計成果物に明記されていること。 ・使用可能文字の規定の変更後 　ToBe規定内文字及びToBe特許庁外字に該当する文字の文字種チェック処理が設計成果物に明記されていること。												
39	167	3.4.3 開発言語の選定ルール	規約1	「表 3.4-5 システム構成要素毎の開発言語と選定指針」で規約とするシステム構成要素は，同表に示す開発言語を使用すること。	1	細則なし	システム構成要素毎に使用する開発言語が設計成果物に明記されており，その内容が「表 3.4-5」に示した開発言語であること。														
			指針1	「表 3.4-5 システム構成要素毎の開発言語と選定指針」で推奨とするシステム構成要素は，同表の「選定指針」に従い開発言語を採用すること。	1	細則なし	システム構成要素毎に使用する開発言語が設計成果物に明記されており，その内容が「表 3.4-5」の選定指針に則っていること。														
			推奨1	「表 3.4-5 システム構成要素毎の開発言語と選定指針」で示す開発言語を推奨する。	1	細則なし	「表 3.4-5」の推奨された開発言語を使用する旨が設計成果物に明記されていること。														

*1: 評価の決定方法については「技術的整合性検証プロセスガイドライン」を参照のこと。
*2: Level-2の評価結果はLevel-1の評価結果より導出する。
　例: Level-1の評価結果に１つでも「不適合」があればLevel-2の評価結果は「不適合」となる。
　導出方法については「技術的整合性検証プロセスガイドライン」を参照のこと。
*3: Level-3の評価結果はLevel-2の評価結果より導出する。
　例: Level-2の評価結果に１つでも「不適合」があればLevel-3の評価結果は「不適合」となる。
　導出方法については「技術的整合性検証プロセスガイドライン」を参照のこと。